```
++++++++++++++++++++++++++
+                        +
+   DOCUMENTATION FOR    +
+                        +
+     N E W D 0 S +      +
+                        +
+ Disk System for TRS-80 +
+                        +
++++++++++++++++++++++++++
```


USE OF THE FOLLOWING PROGRAMS WILL REQUIRE PURCHASE OF RADIO SHACK'S TRSDOS DISK OPERATING SYSTEM, V.2.1 OR LATER. THE EDTASM MODIFICATIONS IN NEWDOS+ WILL REQUIRE PURCHASE OF THE RADIO SHACK EDITOR/ASSEMBLER PACKAGE.

NEWDOS+ INFORMATION -- 03/11/79

HEREIN ARE COMMENTS ON THE CHANGES MADE TO THE TRS-80 DOS SYSTEM BY
APPARAT, INC., 6000 E. EVANS AVE., DENVER CO 80222 (303/708-7275)

****
WITH THESE CHANGES, APPARAT DOES NOT PROVIDE COPIES OR EXTRACTS OF THE ORIGINAL DOCUMENTATION FOR BASIC, FOR DOS OR FOR THE EDITOR-ASSEMBLER. THE ORIGINAL RELEASES OF THESE PROGRAMS ARE COPYRIGHTED BY OTHERS AND SOLD BY RADIO SHACK. THE USERS/BUYERS OF THIS UPDATED SYSTEM ARE ASSUMED TO HAVE PURCHASED THESE ORIGINAL PROGRAMS FROM RADIO SHACK, THEREBY OBTAINING THE ORIGINAL DOCUMENTATION AND IMPLICITLY PAYING THE ROYALTIES DUE. APPARAT'S REFUSAL TO SUPPLY COPIES OR EXTRACTS OF THE ORIGINAL DOCUMENTATION IS ITS SOLE EFFORT TO ASSURE ROYALTIES ARE BEING PAID.

APPARAT, INC.

1. PROVIDES THIS SYSTEM SOLELY IN AN 'AS IS' CONDITION.
2. DOES NOT GUARANTEE THAT, AND PROBABLY DOES NOT KNOW WHETHER, ANY PARTICULAR PROGRAM/APPLICATION WILL FUNCTION PROPERLY OR SATISFACTORILY WITH THIS SYSTEM.
3. REQUIRES THAT THE USER THOROUGHLY TEST THIS SYSTEM BEFORE USING IT WITH VALUED DATA OR APPLICATIONS.
4. IS NOT RESPONSIBLE FOR DAMAGES CAUSED BY USE OF THIS SYSTEM.
5. DOES NOT PROVIDE MAINTENANCE, UPGRADE OR CONSULTATION SERVICES AS PART OF THIS OFFERING.
6. HOPES THE SYSTEM INFORMATION PROVIDED HEREIN IS ACCURATE BUT CANNOT GUARANTEE IT.

****

EACH CHANGE IS HEADED BY A LINE OF ASTERISKS, CONTAINING THE CHANGE ID.

THE HANDLING OF THE DIRECTORIES IS ESSENTIALLY THE SAME AS IN DOS 2.1.

THIS SYSTEM CONSISTS OF THE FOLLOWING MODULES, WHICH IF NOT NOTED AS NEW, ARE SIMPLY THE ORIGINAL DOS 2.1 MODULES WITH CORRECTIONS AND ENHANCEMENTS:

1. DIR/SYS — 2 GRANULES. DISKETTE DIRECTORY.
2. BOOT/SYS — 1 GRANULE. BOOTSTRAPS THE SYSTEM.
3. SYS0/SYS — 1 GRANULE. RESIDENT MODULE. HANDLES DISK I/O, CLOCK INTERRUPTS, LOAD OF SYSTEM OVERLAY MODULES, ETC.
4. SYS1/SYS — 1 GRANULE. INTERROGATES DOS COMMANDS.
5. SYS2/SYS — 1 GRANULE. OPEN EXECUTOR AND OTHER FUNCTIONS.
6. SYS3/SYS — 1 GRANULE. HANDLES CLOSE AND KILL. CONTAINS COPY'S FORMATTING CODE.
7. SYS4/SYS — 1 GRANULE. DISPLAYS SYSTEM ERROR MSGS.
8. SYS5/SYS — 1 GRANULE. 'DEBUG' AND CMD"D" EXECUTOR.
9. SYS6/SYS — 3 GRANULES. DOS COMMAND EXECUTOR, EXCEPT COPY AND DEBUG.
10. FORMAT/CMD — 3 GRANULES. 'FORMAT' EXECUTOR.
11. COPY/CMD — 1 GRANULE. NEW MODULE. EXECUTES DOS COPY COMMAND WITH SINGLE-DRIVE AND 'BACKUP' CAPABILITIES.
12. BASIC/CMD — 4 GRANULES. EXECUTES BASIC'S LEVEL III FUNCTIONS.
13. SYS13/SYS — 1 GRANULES. NEW MODULE. DISPLAYS BASIC'S ERROR MESSAGES. MUST DE IN SYSTEM WHENEVER BASIC IS ACTIVE.
14. SYS12/SYS — 1 GRANULE. NEW MODULE. EXECUTES BASIC DIRECT COMMAND 'REF'. NEED NOT BE IN SYSTEM IF REF WILL NEVER BE EXECUTED.

15. SYS11/SYS 1 GRANULE. NEW MODULE. EXECUTES BASIC DIRECT COMMAND 'RENUM'. NEED NOT BE IN SYSTEM IF RENUM WILL NEVER BE EXECUTED.
16. DIRCHECK/CMD 3 GRANULES. NEW MODULE. MAKES CHECKS AND LIST/ PRINTS DIRECTORY CONTENTS.
17. EDTASM/CMD 5 GRANULES. EDITOR-ASSEMBLER FOR Z-80 CODE. SOURCE AND OBJECT CODE FROM/TO DISK OR TAPE.
18. DISKDUMP/BAS 2 GRANULES. DUMPS DISK FILES TO DISPLAY OK LINE PRINTER.
19. BASIC1/CMD PROVIDES A LEVEL-I-IN-LEVEL-II CAPABILITY.
20. LV1DSKSL/CMD ALLOWS STORAGE AND RETRIEVAL OF LEVEL I PROGRAMS BUT NOT DATA TO/FROM DISK.
21. SUPERZAP/BAS BASIC PROGRAM THAT ALLOWS INSPECTION AND MODIFICATION OF EITHER DISK OR MAIN MEMORY. DISK OPS ARE NOT FILE ORIENTED.
22. SUPERZAP/COM BASIC PROGRAM SERVING AS SUPERZAP DOCUMENTATION.

########## REDUCED-SIZE OPERATING SYSTEMS CAN BE CREATED BY 'COPY'ING FULL NEWDOS DISKETTE, THEN 'KILL'ING UNWANTED FILES (SEE FOLLOWING).

A MINIMUM SYSTEM TO HANDLE OPEN'S AND CLOSE'S WILL CONSIST OF 10 GRANULES (BOOT, DIR, SYS0-SYS4)
IF DEBUG TO BE USED, ADD SYS5.
IF DOS COMMANDS (DIR, CLOCK, APPEND, DATE, ETC) TO BE EXECUTED, ADD SYS6 (SYS5 NOT REQUIRED).
IF DOS COMMAND COPY TO BE EXECUTED, ADD COPY/CMD. (SYS5, SYS6 NOT REQUIRED) IF FORMAT TO BE EXECUTED, ADD FORMAT/CMD. (SYS5, SYS6, COPY NOT REQUIRED)
IF BASIC TO BE ENTERED, ADD BASIC/CMD AND SYS13. ONCE BASIC IS LOADED, IT DOES NOT HAVE TO REMAIN PART OF THE SYSTEM DISKETTE CONTENTS (BUT SYS13 DOES). SYS5 NOT NEEDED IF CMD"D" NOT USED. SYS6, COPY, FORMAT NOT NEEDED UNLESS ASSOCIATED COMMANDS ARE EXECUTED VIA CMD"XX" FUNCTION.
IF BASIC DIRECT COMMAND REF TO BE EXECUTED, ADD SYS12.
IF BASIC DIRECT COMMAND RENUM TO BE EXECUTED, ADD SYS11.

BASIC, COPY, FORMAT DO NOT HAVE TO RESIDE ON THE SYSTEM DISKETTE ON DRIVE 0; BY APPENDING THE DRIVE # (IE, COPY:2) TO THE COMMAND NAME THE MODULE CAN BE TAKEN FROM THE DISKETTE ON THE SPECIFIED DRIVE.

********* THE SYSTEM CHANGES HEREIN MAKE THIS SYSTEM INCOMPATIBLE WITH THE ORIGINAL DOS 2.1 SYSTEM. NO MODULE OF THIS SYSTEM (SYS0-SYS6, SYS11-SYS13, COPY/CMD, BASIC/CMD OR FORMAT/CMD)
MAY BE USED INTERCHANGEABLY WITH MODULES FROM THE ORIGINAL DOS 2.1 OR ANY OTHER SYSTEM, OR VICE-VERSA. CHANGING SYSTEMS MUST BE ACCOMPLISHED VIA 'RESET' OR 'POWER-ON' TO ASSURE THE PROPER RESIDENT MODULE (SYS0) LOAD.
WHEN COPYING FILES INTO OR OUT OF THIS SYSTEM, THE SOURCE AND/OR THE DESTINATION FILE MAY RESIDE ON AN ALIEN SYSTEM DISKETTE PROVIDED ONE OF THE FOLLOWING EXISTS:
1. THE ALIEN SYSTEM DISKETTE IS NEVER MOUNTED ON DRIVE ZERO.
2. THE COPY COMMAND HAS THE SOURCE FILESPEC PRECEDED BY A $.

THE ORIGINAL DOS 2.1 'BACKUP' MODULE WILL WORK WITH THIS SYSTEM. HOWEVER, AN OPTION OF THIS SYSTEM'S COPY ACCOMPLISHES ESSENTIALLY THE SAME THING.
****

USE OF 'COPY' IS DISCUSSED IN CHANGE # 040. ITS FORMAT #5 REPLACES 'BACKUP'.
USE OF 'REF' IS DISCUSSED IN CHANGE # 045.
USE OF 'RENUM' IS DISCUSSED IN CHANGE # 046.
USE OF 'FORMAT' IS DISCUSSED IN CHANGE # 037.
USE OF 'JKL' OPTION IS DISCUSSED IN CHANGE # 017.
KEYBOARD DEBOUNCE IS DISCUSSED IN CHANGE # 015.
NEW BASIC INVOCATION IS DISCUSSED IN CHANGE # 039.
BASIC'S CMD'XX' FUNCTION IS DISCUSSED IN CHANGE # 035.

BASIC SCROLLING COMMANDS DISCUSSED IN CHANGE # 031.
TRUNCATED BASIC LIST, EDIT AND DELETE COMMANDS ARE DISCUSSED IN CHANGES
047 AND 031.
BASIC OPEN"E" IS DISCUSSED IN CHANGE # 019.
'DISKDUMP/BAS' IMPROVEMENTS DISCUSSED IN CHANGE # 048.
'EDTASM/CMD' IMPROVEMENT DISCUSSED IN CHANGE # 049.
LEVEL I IN LEVEL II OPERATION. CHANGES # 050 AND 051.

****

THE FOLLOWING SHOULD BE NOTED:

1. THIS SYSTEM DOES NOT ADVANCE DATE AT 2400.
2. ALL PASSWORD PROTECTION IS DISABLED. SEE CHANGE #008 FOR PASSWORD ENABLE/DISABLE INSTRUCTIONS.
3. DEVICE COMMAND IS DISABLED.
4. ALL BASIC WRITES TO DISK, VIA PRINT AND PUT COMMANDS ARE VALIDITY READ. USER HAS NO CONTROL OF THIS.
5. ALL BASIC PROGRAMS 'SAVED' TO DISK ARE VALIDITY READ.
6. ALL FILES DUPLICATED BY 'COPY' ARE VALIDITY READ.
7. 'APPEND' NOW WORKS.
8. FILE EOF'S ARE HANDLED CORRECTLY.
9. BASIC'S 'LOC' COMMAND NOW WORKS PROPERLY.
10. DOS CMD 'VERIFY' NOW WORKS PROPERLY.
11. ALL SYSTEM DIRECTORY WRITES ARE NOW VALIDITY READ.
12. BASIC 'LOAD' AND 'SAVE' RUN 0-3O% FASTER, DEPENDING UPON PROGRAM SIZE.
13. DUE TO CORRECTIONS IN SYS3, A NUMBER OF DIRECTORY CLOBBERING SITUATIONS HAVE BEEN ELIMINATED.
14. DISK SPACE ALLOCATION IS 1 GRANULE INCREMENTS, INSTEAD OF 2. THIS INCREASES WRITE OVERHEAD BUT ALLOWS LAST GRANULE ON DISKETTE TO BE USED.
15. BASIC'S OPEN"E" FUNCTION ALLOWS A BASIC PROGRAM TO ADD TO AN EXISTING SEQUENTIAL FILE.
16. THE DISK BACKUP FUNCTION OF SUPERZAP ALLOWS INFINITE RETRY TO RECOVER INFORMATION FROM A BAD SECTOR WHEN 'BACKUP' OR COPY (FULL DISKETTE OPTION) FAIL. IF THAT STILL FAILS, THEN BYPASS THE BAD SECTORS) AND USE SUPERZAP'S 'SCOPY' OPTION TO RECOVER AS MUCH AS POSSIBLE OF THE BAD SECTOR, MOVING IT TO A DIFFERENT SECTOR WHERE IT POSSIBLY CAN BE CORRECTED.
17. THE 'JKL' OPTION ALLOWS THE FULL DISPLAY CONTENTS TO BE SENT TO THE LINE PRINTER, WITHOUT DISTURBING OR ADDING TO THE DISPLAY CONTENTS.
18. LEVEL1-IN-LEVEL2 OPERATION IS AVAILABLE THROUGH 'BASIC1' AND 'LV1DSKSL' DOS COMMANDS. SEE CHANGES 050 AND 051.
19. UNDER OLD DOS, IF NORMAL USER DISK I/O REFERENCES A DRIVE WITHOUT A DISKETTE OR WITH DISKETTE IMPROPERLY MOUNTED, THE SYSTEM MIGHT HANG. THIS SYSTEM WILL PROBABLY CATCH THE CONDITION. THOUGH IT MIGHT TAKE 30 SECONDS.
20. THE OCCURRENCE OF 'LOST DATA' DISK ERRORS IS GREATLY REDUCED.
21. ON EVERY SECTOR READ/WRITE, THE CLOCK LOSES APPROXIMATELY 120MS. IF THIS IS IMPORTANT, SEE CHANGE #007.

****

DOS NOMENCLATURE AND STRUCTURE:

1. SECTOR — UNIT OF 256 BYTES STORED UN DISK.
2. GRANULE — UNIT OF 5 SECTORS. THE BASIC UNIT OF DISK STORAGE ALLOCATION. DIRECTORIES KEEP TRACK OF FREE/ASSIGNED DISK SPACE IN TERMS OF GRANULES.
3. TRACK — UNIT OF 10 SECTORS (2 GRANULES), EQUALS THE # OF DATA BYTES THE DISK READ/WRITE HEAD PASSES OVER ON EACH DISKETTE REVOLUTION.
4. BOOT/SYS — A FILE PLACED UN EVERY DISKETTE BY 'FORMAT'. THIS IS THE BOOTSTRAP ROUTINE INVOKED WHEN RESET OR POWER-ON OCCURS. IT AUTOMATICALLY LOOKS IN THE DIRECTORY FOR SYS0/SYS'S ENTRY. IF PRESENT AND ACTIVE, SYS0/SYS IS LOADED INTO MEMORY AS THE DOS RESIDENT MODULE. OTHERWISE THE SYSTEM HANGS.
5. DIR/SYS — A FILE PLACED ON EVERY DISKETTE BY FORMAT, IT IS REQUIRED FOR PROPER DOS OPERATION. CURRENTLY IT CONSISTS OF 10 SECTORS USED AS FOLLOWS:
    1. SECTOR 0 (CAT SECTOR) CONTAINS:
        1. DISKETTE NAME. (OFFSET = D0-D7 HEX)
        2. LATEST CREATION, BACKUP ON COPY DATE. (OFFSET = D8-DF HEX)
        3. DISKETTE PASSWORD ENCODE. (OFFSET = CE-CF HEX)
        4. DOS 'AUTO' INFORMATION. (OFFSET E0-FF HEX)
        5. TRACK LOCKOUT INFO. OFFSET 60 HEX AND UP FOR AS MANY BYTES AS THE DISKETTE HAS TRACKS. FC HEX VALUE MEANS TRACK NOT LOCKED OUT; FF HEX VALUE MEANS TRACK IS LOCKED OUT.
        6. GRANULE FREE/ASSIGNED STATUS. OFFSET 00 HEX AND UP FOR AS MANY BYTES AS THE DISKETTE HAS TRACKS. EACH BYTE CONTAINS INFORMATION FOR THE TWO GRANULES ON THAT TRACK. 1ST 6 BITS SHOULD ALWAYS = 1. 7TH BIT IS FOR THE 2ND GRANULE OF THE TRACK (SECTORS 5-9). THE 8TH BIT IS FOR THE 1ST GRANULE OF THE TRACK (SECTORS 0-4). FOR THESE TWO BITS THE CORRESPONDING GRANULE IS FREE IF THE BIT = 0; IS ALLOCATED TO A FILE IF = 1.
    2. SECTOR 1 (HIT SECTOR) CONTAINS A 1 BYTE HASH CODE FOR EACH ACTIVE DIRECTORY ENTRY (4TH BITS 1ST BYTE = 1) IN SECTORS 2-9. THUS ALLOWING RAPID ACCESS TO FILES. THE HASH CODE IS BASED ON THE CONTENTS OF FPDE BYTES 6 - 16. THE RELATIVE POSITION IN THE HIT SECTOR OF A DIRECTORY ENTRY'S HASH CODE IS EQUAL TO THAT ENTRY'S DEC. THE OTHER 192 BYTES OF THE SECTOR ARE NOT CURRENTLY USED AND SHOULD ALWAYS = 0.
    3. SECTORS 2-9 EACH CONTAIN SPACE FOR EIGHT 32 BYTE DIRECTORY ENTRIES FPDE OR FXDE). 1ST TWO ENTRIES OF EACH SECTOR ARE RESERVED FOR THE SYSTEM. NORMALLY THE DIRECTORY IS LOCATED ON DISKETTE'S RELATIVE TRACK 11 HEX.
6. FPDE — FILE PRIMARY DIRECTORY ENTRY. EACH FILE, WHEN CREATED, IS ASSIGNED A DIRECTORY ENTRY SOMEWHERE IN DIRECTORY SECTORS 2-9. THIS ENTRY CONTAINS:
    1. 1ST BYTE, 1ST BIT = 0, INDICATING FPDE, VICE FXDE.
    2. 1ST BYTE, 2ND BIT = 1, IF A SYSTEM FILE.
    3. 1ST BYTE, 4TH BIT = 1 IF THIS ENTRY IS ASSIGNED TO A FILE AND HAS A NON-ZERO HASH CODE IN THE HIT SECTOR. THE OFFSET IN THE HIT SECTOR OF THIS ENTRY'S HASH CODE = THIS ENTRY'S DEC. IF THIS BIT = 0, THE DIRECTORY ENTRY IS AVAILABLE FOR REASSIGNMENT; I.E., IF FIRST HEX DIGIT ON SUPERZAP'S DUMP LINE IS 0.
    4. 1ST BYTE, 5TH BIT = 1 1F A FILE HAS THE INVISIBLE ATTRIBUTE.
    5. 1ST BYTE, 6-8TH BITS = PROTECTION LEVEL:
        7 = NO ACCESS
        6 = EXECUTION ACCESS ONLY
        5 = READY EXECUTE ACCESSES
        4 = WRITE, READY EXECUTE
        3 = (UNUSED)

2 = RENAME, WRITE, READ, EXECUTE
1 = KILL, RENAME, WRITE, READ, EXECUTE
0 = NO RESTRICTIONS

6. 2ND BYTE = 0, USE UNKNOWN.
7. 3RD BYTE = 0, USE UNKNOWN.
8. 4TH BYTE = 0-255 END-OF-FILE RELATIVE POSITION IN THE EOF SECTOR.
9. 5TH BYTE = 0-255 LOGICAL RECORD LENGTH, 0 = 256. WHEN FILE IS CREATED VIA A 4420H VECTOR CALL THE VALUE FROM REGISTER B IS STORED HERE. WHEN AN EXISTING FILE IS OPENED, EVEN AS A NEW OUTPUT FILE THIS VALUE IS NOT UPDATED. THIS VALUE IS NEVER USED. THE VALUE STORED IN FCB+9 AT OPEN TINE IS THAT FROM REGISTER B NOT FROM THE FPDE.
10. 6TH-13TH BYTES = FILE NAME, PADDED ON RIGHT WITH BLANKS IF NECESSARY.
11. 14TH-15TH BYTES = FILE NAME MODIFIER, PADDED ON RIGHT WITH BLANKS AS NECESSARY.
12. 17TH-18TH BYTES = ENCODE OF UPDATE PASSWORD.
13. 19TH-20TH BYTES = ENCODE OF ACCESS PASSWORD.
14. 21ST-22ND BYTES = RELATIVE SECTOR # WITHIN THE FILE OF EOF. HOWEVER, IF THE 4TH BYTE IS NON-ZERO THIS VALUE IS RELATIVE EOF SECTOR + 1. THIS SYSTEM MAINTAINS THE EOF WITHIN THE FPDE COMPATIBLE WITH THAT OF TRSDOS 2.1 SO THAT A FILE MAY BE USED INTERCHANGEABLY BETWEEN THE TWO SYSTEMS. WHEN A FILE IS OPEN, THE EOF VALUE IN THE FCB IS MAINTAINED DIFFERENTLY IN THE TWO SYSTEMS.
15. 23RD-30TH BYTES = FOUR 2 BYTE PAIRS (EXTENT ELEMENTS), EACH SPECIFYING A CONTIGUOUS AREA OF THE DISKETTE ASSIGNED TO THIS FILE. THE FORMAT OF AN EXTENT ELEMENT IS:
    1. 1ST BYTE;
        1. 255 = END OF EXTENTS.
        2. 254 = 2ND BYTE CONTAINS THE DEC FOR THE NEXT FXDE ASSIGNED TO THIS FILE TO CONTAIN ADDITIONAL EXTENT INFORMATION.
        3. 0 - 34 = RELATIVE TRACK ON DISKETTE FOR START OF AREA. IF YOUR DISKETTES HAVE MORE THAN 35 TRACKS, THIS VALUE CAN COVER 0 - 253; RANGE.
    2. 2ND BYTE (WHEN 1ST BYTE = 254)
        1. LEFT 3 BITS = # GRANULES (0-7) OFFSET FROM START OF TRACK TO START OF AREA.
        2. RIGHT 5 BITS = #-1 OF CONTIGUOUS GRANULES ASSIGNED TO THIS AREA.
16. 31ST-32ND BYTES CONTAIN AN EXTENT ELEMENT WHOSE 1ST BYTE IS EITHER 255 OR 254.

7. FXDE — FILE EXTENDED DIRECTORY ENTRY. WHEN A FILE HAS MORE THAN 4 SPACE AREAS ASSIGNED, THE ADDITIONAL EXTENT ELEMENTS ARE CONTAINED IN FXDE'S ASSIGNED TO THE FILE. THE FORMAT OF A FXDE IS:
    1. 1ST BYTE, 1ST BIT - 1 TO INDICATE FXDE, VICE FXDE.
    2. 1ST BYTE, 4TH BIT. SEE FPDE.
    3. 2ND BYTE = DEC FOR PREVIOUS FXDE OR FPDE OF THIS FILE. THIS IS A BACKWARD CHAIN. THE PREVIOUS ENTRY'S 31ST BYTE WILL BE 254r AND THE 32ND BYTE WILL, CONTAIN THE DEC OF THIS FXDE.
    4. BYTES 3-22 ARE UNUSED AND SHOULD = 0.
    5. BYTES 23-32 ARE AS DEFINED FOR FPDE.
8. EXTENT ELEMENT (SEE FPDE, BYTES 23-30).
9. DEC — DIRECTORY ENTRY CODE. A ONE-BYTE CODE USED TO SPECIFY ONE OF THE 64 DIRECTORY ENTRIES IN DIRECTORY SECTORS 2 - 9. THE FORMAT IS:

    RRR00SSS WHERE:

    SSS+2 = RELATIVE SECTOR IN DIRECTORY
    RRR = RELATIVE 32 BYTE DIRECTORY ENTRY IN THAT SECTOR.

10. GAT — GRANULE ALLOCATION TABLE. REFERS TO DIRECTORY RELATIVE SECTOR 0.
11. HIT — HASH INDEX TABLE. REFERS TO DIRECTORY RELATIVE SECTOR 1.
12. VICE — MEANS 'INSTEAD OF' OR 'IN PLACE OF'.
13. FCB — FILE CONTROL BLOCK. ALSO KNOWN AS AN IOB (INPUT/OUTPUT BLOCK). AT OPEN TIME (CALL TO DOS 4420H OR 4424H), THE CALLER PROVIDES IN REGISTER DE THE ADDRESS OF A 32 BYTE MAIN MEMORY AREA FOR USE BY THE SYSTEM WHILE THE FILE IS OPEN. THE USER MUST HAVE PLACED THE FILE-SPEC INTO THE FCB'S 1ST BYTES, AND CLOSE WILL ATTEMPT TO PUT IT BACK THERE WHEN DONE. WHILE THE FILE IS OPEN, THE FORMAT FOR THE 32 BYTE FCB IS:
    1. 1ST BYTE, 1ST BIT = 1, FILE IS OPEN; = 0, FILE IS CLOSED.
    2. 2ND BYTE, 1ST BIT
       = 1, EITHER SINGLE BYTE OPERATIONS OR LOGICAL RECORD OPERATIONS (REC LENGTH IN 10TH BYTE) BEING DONE ON THIS FILE. 'NEXT' VALUE IS MAINTAINED AT THE NEXT BYTE TO BE READ/WRITTEN.
       = 0, SECTOR OPERATIONS; CALLS TO SYSTEM ARE BY FULL 256 BYTE SECTOR. 'NEXT' VALUE IS MAINTAINED AT START OF NEXT SECTOR TO BE READ/WRITTEN. THIS BIT IS SET = 1 AT OPEN TIME IF REGISTER B NOT = 0. IT IS ALSO SET = 1 WHENEVER ONE OF THE ROM (LEVEL II BASIC) SINGLE BYTE READ/WRITE ROUTINES ARE CALLED WITH REGISTER DE CONTAINING THE FCB ADDR.
    3. 2ND BYTE, 2ND BIT
       = 0, EOF IS TO BE SET = TO 'NEXT' ON EVERY WRITE OPERATION.
       = 1, EOF IS TO BE SET = TO 'NEXT' ONLY FOR THOSE WRITE OPERATIONS RESULTING IN 'NEXT' EXCEEDING THE OLD EOF.
    4. 2ND BYTE, 3RD BIT
       = O, BUFFER CONTAINS THE CURRENT SECTOR.
       = 1, BUFFER DOES NOT CONTAIN THE CURRENT SECTOR.
    5. 2ND BYTE, 4TH BIT
       = 0, BUFFER DOES NOT CONTAIN UPDATED DATA NOT YET SENT TO THE FILE.
       = 1, BUFFER DOES CONTAIN UPDATED DATA NOT YET SEND TO THE FILE.
       AT CLOSE TIME, IF THIS BIT AND 2ND BYTE, 1ST BIT BOTH = 1, THE RESIDUAL SECTOR IN THE BUFFER IS AUTOMATICALLY WRITTEN TO DISK.
    6. 2ND BYTE, 6-8TH BITS = PROTECTION CODE (SEE FPDE, 1ST BYTE 6 - 8TH BITS) IF PASSWORD CHECKING ENABLED AND ACCESS, VICE UPDATE, PASSWORD SPECIFIED IN THE FILESPEC AT OPEN TIME.
    7. 3RD BYTE, SET = 0 AT OMEN TIME. DON'T KNOW WHAT IT'S USED FOR.
    8. 4-5TH BYTES CONTAIN THE BUFFER ADDRESS SUPPLIED IN REG HL BY CALLER AT OPEN TIME.
    9. 6TH BYTE LOW ORDER BYTE OF 3 BYTE 'NEXT' VALUE. THIS IS THE RELATIVE POSITION WITHIN SECTOR VALUE. SEE DISCUSSION FOR 11-12TH BYTES BELOW.
    10. 7TH BYTE RELATIVE DRIVE # OF DRIVE CONTAINING DISKETTE CONTAINING THE FILE AT OPEN TIME.
    11. 8TH BYTE DEC OF FILE'S FPDE.
    12. 9TH BYTE LOW ORDER BYTE OF EOF. SEE DISCUSSION OF FCB, 13-14TH BYTES.
    13. 10TH BYTE 0-256 LOGICAL RECORD LENGTH (LRL) FOR LOGICAL RECORDS OF THIS FILE. 0 = 256. THIS VALUE IS SUPPLIED IN REGISTER B BY THE CALLER AT OPEN TIME. IF NOT = 0 AT OPEN TIME, FCB 1ST BYTE, 1ST BIT IS SET = 1 AND DOS SECTOR READ/WRITE CALLS MUST CONTAIN IN REGISTER HL THE ADDRESS OF THE LOGICAL RECORD TO BE MOVED TO THE FCB'S BUFFER (WRITE) OR FILLED FROM THE FCB'S BUFFER (READ).
    14. 11-12TH BYTES 12TH, 11TH AND 5TH BYTES FORM A 3 BYTE RELATIVE OFFSET IN FILE TO THE NEXT BYTE TO BE PROCESSED, EITHER INPUT

OR OUTPUT. FOR BYTES OPERATIONS (2ND BYTE, 1ST BIT = 1) THIS SYSTEM MAINTAINS THIS 3 BYTE VALUE EXACTLY WHEREAS THE ORIGINAL DOS SYSTEM MAINTAINED THIS VALUE HIGH BY 256 CAUSING SOME PROBLEMS IN 'COPY'.
FOR FULL SECTOR OPERATIONS, BOTH THE ORIGINAL DOS AND THIS SYSTEM ADVANCE 'NEXT BY 256 ON EACH SUCCESSFUL READ OR WRITE TO DISK, EXCEPTING THAT IN THIS SYSTEM, 'NEXT' IS NOT ADVANCED ON FULL SECTOR WRITE IF THE FCB'S 6TH BYTE NOT = 0.

15. 13-14TH BYTES. THE 14TH, 13TH AND 8TH BYTES FORM 3 BYTE RELATIVE BYTE OFFSET IN THE FILE OF END-OF-FILE (THE 1ST BYTE BEYOND THE FILE'S LAST DATA BYTE). THIS VALUE IS INITIALIZED FROM THE FPDE AT OPEN TIME, AND IS UPDATED AT SECTOR, LOGICAL RECORD OR BYTE WRITE TIME UNDER CONTROL OF FCB, 2ND BYTE, 2ND BIT. THIS SYSTEM MAINTAINS THE EOF IN THE FCB EXACTLY CURRENT WHILE THE ORIGINAL DOS SYSTEM MAINTAINED IT HIGH BY 256 WHENEVER FCB 6TH BYTE NOT = 0. FOR COMPATIBILITY, THIS SYSTEM, AT CLOSE TIME WHEN THE EOF VALUE IN THE FPDE IS UPDATED, MAKES THE EOF IN THE FPDE EXACTLY AS FOR THE ORIGINAL DOS SYSTEM.
16. 15-16TH BYTES. IDENTICAL TO 23-24TH BYTES OF FPDE.
17. 17-32TH BYTES. FOUR 4 BYTE ELEMENTS OF THE FORM:
    1. 1-2ND BYTES, RELATIVE GRANULE OFFSET WITHIN THE FILE TO THE START OF THE DISK AREA WHOSE EXTENT ELEMENT FOLLOWS.
    2. 3-4TH BYTES, THE EXTENT ELEMENT OF ONE OF THE AREAS ASSIGNED TO THE FILE (EXCEPT THE FIRST). SEE FPDE, BYTES 23-30.

THESE FOUR ELEMENTS ALLOW THE SYSTEM TO REDUCE DIRECTORY ACCESS DURING RANDOM FILE OPERATIONS BY REMEMBERING THE MOST 4 RECENT AREAS (OTHER THAN THE FIRST) ACCESSED, UNDER THE ASSUMPTION THAT A SECTOR FROM ONE OF THOSE AREAS WILL BE THE NEXT ACCESSED OR WRITTEN.

14. EOF END-OF-FILE. THE RELATIVE BYTE ADDRESS+1 WITHIN THEN FILE OF THE LAST BYTE OF THE FILE.

***** 001 **

DOS COMMAND 'DIRCHECK'

THE DIRCHECK/CMD MODULE TESTS AND LISTS THE TARGET DISKETTE'S DIRECTORY.

TO THE QUERY 'OUTPUT TO PRINTER', REPLY Y FOR YES OR N FOR NO (OUTPUT TO DISPLAY).
TO THE QUERY 'WHICH DRIVE CONTAINS TARGET DISKETTE' REPLY A DECIMAL DIGIT 0 TO 3.

COMPLAINTS ABOUT THE DIRECTORY ARE LISTED FIRST. IF A NUMBER IS GIVEN IT IS IN HEXADECIMAL FOR USE IN DIRECTORY REPAIR VIA 'SUPERZAP'. DO NOT TRY TO REPAIR A BAD DIRECTORY UNLESS YOU KNOW WHAT YOU ARE DOING!!!!!!!
THE NEXT BEST THING IS TO TRY TO EXTRACT VALUED FILES VIA COPY AND THEN RE-FORMAT THE DISKETTE HAVING THE BAD DIRECTORY.

IF THE COMPLAINT IS ABOUT A DIRECTORY ENTRY FOR A FILE, EITHER THE PRIMARY OR AN EXTENDED ENTRY, THE HEXADECIMAL CODE IS THE DEC FOR THE FILE'S FPDE.
WHEN THE COMPLAINT DEALS WITH A FILE EXTENDED DIRECTORY ENTRY BUT DOES NOT SPECIFY THE FILE NAME/TYPE, THE HEXADECIMAL CODE IS THE DEC FOR THE FXDE ITSELF.
WHEN THE COMPLAINT DEALS WITH A HIT SECTOR BYTE, THE HEXADECIMAL CODE IS THE RELATIVE ADDR OF THAT BYTE IN THE HIT SECTOR.
WHEN THE COMPLAINT DEALS WITH A GAT SECTOR BYTE, THE HEXADECIMAL CODE IS THE RELATIVE ADDR OF THAT BYTE IN THE GAT SECTOR.
WHEN THE COMPLAINT DEALS WITH A GRANULE, THE HEXADECIMAL VALUE IS THE RELATIVE GRANULE NUMBER WITHIN THE DISKETTE.
WHEN THE COMPLAINT DEALS WITH A TRACK, THE HEXADECIMAL VALUE IS THE RELATIVE TRACK NUMBER WITHIN THE DISKETTE.
'DIRCHECK' ALSO DEMANDS THAT THE 1ST 2 BYTES OF THE DISKETTE'S 1ST SECTOR (BOOT/SYS'S 1ST SECTOR) CONTAIN 00 HEX AND FE HEX RESPECTIVELY. THE 3RD BYTE MUST CONTAIN THE RELATIVE # OF THE TRACK CONTAINING THE DIRECTORY,

THE FILES ARE NEXT LISTED, WITH NUMERIC VALUES IN DECIMAL.

1.	S	SYSTEM FILE.
2.	I	FILE HAS INVISIBLE ATTRIBUTE.
3.	P=NNN	FILE HAS PROTECTION LEVEL NNN, AND BOTH UPDATE AND ACCESS PASSWORDS ARE NON-BLANK.
4.	EOF=SSS/BBB	SSS = THE RELATIVE SECTOR WITHIN THE FILE. BBB = THE RELATIVE BYTE WITHIN THE SECTOR.
5.	NNN EXTS	THE # OF EXTENT ELEMENTS WITH 1ST BYTE < 254.
6.	NNN SECTORS	THE NUMBER OF SECTORS ALLOCATED TO THIS FILE.

TO THE QUERY 'START PROGRAM ANEW?', REPLY Y FOR YES OR N FOR NO (EXITS TO DOS).

DURING DISPLAY/LISTING, PRESSING:

'BREAK'	-	DISPLAYING/LISTING WILL PAUSE AT END OF CURRENT LINE OR LINE GROUP.
'ENTER'	-	RESTARTS PAUSED DISPLAYING/LISTING.
'UP-ARROW'	-	TERMINATE DISPLAYING/LISTING.

***** 002 ***

CHANGE TO CAUSE DISK SEEK AND RESTORES TO CONSTANTLY RESELECT THE CURRENT DRIVE WHILE WAITING FOR DISK CONTROLLER TO CONCLUDE THE SEEK/RESTORE. IN THE ORIGINAL SYSTEM, THERE IS INSUFFICIENT DRIVE 'READY' TIME TO MOVE THE ARM FROM TRACK 00H TO TRACK 22H AND HAVE A FULL 20OMS TO READ THE FARTHEST SECTOR ON THE TRACK. THIS IS DUE TO 'READY' BEING ON ONLY APPROX 2.4 SECS FROM LAST SELECT (WRITE TO 37E1H). THE TIME EXPIRES AFTER A 1 SECOND WAIT FOR DRIVE DEFAULT-UP-T0-SPEED AND THEN 34 40MS STEPS OF THE DISK ARM. DOS-BASIC 'GET' AND 'PUT' AVOIDED THIS PROBLEM BY DOING AN EXTRA SEEK VIA CALL AT 62ABH.
ANYWAY, THIS CHANGE KEEPS THE DRIVE READY FOR THE ENTIRE SEEK/RESTORE (EVEN IF THE ARM MUST STEP 70 TRACKS) PLUS APPROX. 2.4 SECOND BEYOND THAT.

SYS0 4869 AND OTHERS
SYS3, FORMAT PART
FORMAT 5750 5764, 556D

***** 003 **

ENABLES DEVICE-NOT-READY CONDITION TO BECOME A VALID ERROR IN SYS0'S READ/WRITE SECTOR ROUTINES, WHEREAS BEFORE IT WAS GOING UNNOTICED AND CAUSING CALLING ROUTINES TO ASSUME OPERATION COMPLETED WITHOUT ERROR.

SYS0 46AF
******** INCORPORATED IN CHANGE #007

***** 004 **

DIRECT ADDRESSING, VICE INDIRECT, REQUIRED HERE.
THIS ERROR CAUSES REAL TROUBLE FOR PROGRAMS THAT WANT TO USE 'BREAK' AS IT CAUSES THE SYSTEM TO THINK 'DEBUG' IS ACTIVATED WHEN IT HAS NOT BEEN. 'EDTASM' MUST RUN WITH INTERRUPTS OFF BECAUSE OF THIS.

SYS1, 4E28

***** 005 **

CODE FOR SHIFTING DOWN OLDER USED EXTENT REFERENCES IN THE FCB IS JUST PLAIN WRONG.

SYS0, 497A

***** 006 **

NC STATE ALWAYS SET HERE. USE NZ INSTEAD, WHICH MEANS >= EOF.

SYS0, 47F9
******** INCORPORATED IN CHANGE #015

***** 007 ***

CHANGE TO:

1. FORCE VALIDITY READ AFTER EVERY SECTOR WRITE WHEN:
   1. BIT 7, FCB 2ND BYTE = 1 (BYTE OPS)
   2. BASIC IS WRITING TO DATA FILLS
   3. ALL SYSTEM WRITES NOT FOLLOWED BY A SUBSEQUENT READ.
2. DISABLE INTERRUPTS DURING SECTOR READ/WRITE OPERATIONS. THIS CAUSES THE CLOCK TO LOSE APPROX 120MS EACH I/O. IF A MORE ACCURATE CLOCK IS REQUIRED FOR USER OPERATIONS, THIS CAN BE REDUCED TO ABOUT A 10MS CLOCK LOSS PER I/O WHILE HAVING I/O'S TAKE 5-10% LONGER BY CHANGING SYS0, 4692. SYS0, 4692 HEX IS LOCATED AS RELATIVE BYTE 4A HEX OF RELATIVE SECTOR 4 OF FILE SYS0/SYS, AND IF FILE SYS0/SYS'S FPDE 23-24TH BYTES = 0022 HEX, THEN THE TARGET BYTE IS RELATIVE BYTE 4A HEX IN RELATIVE SECTOR 9 OF RELATIVE TRACK 0 ON THE DISKETTE. THE BYTE PRECEDING THE TARGET BYTE = 37 HEX AND THE BYTE FOLLOWING = 36 HEX.
   1. SET TARGET BYTE = 00 HEX IF A MORE ACCURATE CLOCK BUT SLOWER I/O IS WANTED.
   2. SET TARGET BYTE = F3 HEX IF CLOCK VALUE NOT IMPORTANT AND THE FASTER I/O WANTED. THIS SYSTEM IS INITIALLY SET TO THIS STATE.
      AFTER CHANGING THIS BYTE, DO 'RESET'.
3. CATCH DEVICE-NOT-READY EVEN THOUGH CONTROLLER REMAINS BUSY.
4. DELAY RE-ENABLING INTERRUPTS UNTIL TERMINATING CONDITIONS HAVE BEEN EXTRACTED FROM CONTROLLER. DON'T KNOW IF THIS WAS AN ERROR, BUT IT SEEMS SENSIBLE.

****** HOPEFULLY, THIS WILL REDUCE DISK ERRORS ....

SYS0, 4860, 4671, 468F, 45BF
SYS2 514E
SYS3 5014
SYS6 5445
BASIC 62BF

***** 008 **

DISABLE ALL PASSWORD CHECKS
PROTECTION CODE IS DISABLED. PROGRAMS OPERATE AS IF THEY HAD NO PASSWORDS AT ALL. THIS IS NOT AN ERROR CORRECTION BUT A CONVENIENCE. THIS PATCH NEEDED WHEN ACCESS/UPDATE NEEDED TO SYSTEM PROGRAMS BY METHOD OTHER THAN 'SUPERZAP'. ONLY ONE BYTE IS CHANGED, THAT LOCATED AT SYS2, 4EAC. WITH PASSWORD CHECKING ENABLED, THAT BYTE = 28 HEX. WITH PASSWORD CHECKING DISABLED THE BYTE = 18 HEX. THIS BYTE CAN BE ALTERED VIA 'SUPERZAP'. THE BYTE IS LOCATED IN FILE SYS2/SYS, RELATIVE SECTOR 1, RELATIVE BYTE 63 HEX. IF FILE SYS2/SYS HAS ITS FPDE, 23-24TH BYTES = 1020 HEX, THEN THE BYTE IS AT RELATIVE POSITION 63 HEX WITHIN RELATIVE SECTOR 6 WITHIN RELATIVE TRACK 10 HEX OF THE SUBJECT DISKETTE. AS A CHECK, THE PRECEDING BYTE = E1 HEX AND THE FOLLOWING BYTE = 21 HEX.

SYS2, 4EAC

***** 009 ***

SYS4: B AND C REGS LOADED TO LATE FOR DIRECTORY SECTOR READ AT 4EC4H.

SYS4, 4E86

***** 010 ***

SYS0: PROTECTION CHECK DURING LOAD/EXEC FUNCTION IS THAT FOR 'READ', NOT 'EXEC'
ALSO ENABLE CALL OF FILE-ACCESS-INHIBITED ERROR FOR LOAD/EXEC REGARDLESS OF 'DEBUG'
STATUS.

SYS0, 4C2E

***** 011 ***

HAVE SYS0 ALLOCATE IN UNITS OF 1 GRANULE, VICE 2, WHICH ALLOWS USE OF LAST GRANULE
ON DISKETTE. THIS WILL INCREASE WRITE OVERHEADS.

SYS0, 4A17

***** 012 ***

CHANGE TO ALLOW 'CLOSE' FUNCTION IN SYS3 TO DEALLOCATE THE UNUSED GRANULE WHEN THE
LOGICAL END OF FILE IS THE FIRST BYTE OF THE GRANULE. THIS ASSUMES THAT THE EOF
BYTE IS HOT A FART OF THE FILE.

SYS3, 4E80, 501B

***** 013 ***

'LOAD' IN BASIC SETS 'END+1 OF TEXT' IN 40F9,A ONE BYTE TOO HIGH. TENDS TO MAKE
BASIC PROGRAM MODULES 1 BYTE TOO LONG, BUT HAS NO NOTICEABLE ERRONEOUS EFFECT.

BASIC: 5FDC

***** 014 ***

DOS SYS0, INIT INITIALIZES 4158-43FF AND THEN, THROUGHOUT SUBSEQUENT SYSTEM
OPERATION, NEVER USES IT. EXCEPT FOR UNEXPLAINED REFERENCE TO 4359 IN SYS1 A
REFERENCE TO 43C0-FF BY SYS6 FOR 'DEVICE', AND AN UNINTENTIONAL REFERENCE TO 43FFH
AS A RESULT OF INSTRUCTIONS IN SYS0, 46C1-CE.
I HAVE CRITICAL NEED FOR A DOS RESIDENT PATCH AREA.
I WILL GAMBLE THAT THIS AREA IS TRULY AVAILABLE AND USE IT. 'DEVICE' COMMAND IS
DISABLED IN THIS SYSTEM.

************* EXTREME CAUTION!!!!!!!!! *******
SYS0, 4E20
SYS6, 531D
**************** SEE SUBSEQUENT PATCHES FOR THIS AREA'S USE.

***** 015 ***

DOS MISUSES, AT TIMES, THE CONCEPT OF EOF (FPDE+15H, 14H, 03H)(FCB+0DH, 0CH, 08H) AND OF CURRENT POSITION WITHIN THE FILE (FCB+0BH, 0AH, 05H).
THESE TWO 3 BYTE VALUES REPRESENT RELATIVE BYTE POSITIONS WITHIN THE FILE OF:

1. THE 1ST BYTE AFTER THE LOGICAL END OF FILE.
2. THE NEXT BYTE TO BE PROCESSED EITHER BY READ OR WRITE.

THESE CHANGES ARE TO:

1. ALLOW PROPER OPERATION UNDER THIS CONCEPT.
2. YET. PRESERVE EOF VALUE IN FPDE IN DOS ERRONEOUS STATE SO FILES CAN BE USED IN EITHER THE 'ORIGINAL' DOS SYSTEM OR MY 'ALTERED' SYSTEM. BUT NOT A MIXTURE OF BOTH.

SYS2, 4FE8
SYS3, 4E19
SYS0, 4864, 47F6, 47C5, 47CC

***** 016 ***

IF THE NEW KEY PRESSED IS THE SAME AS THE LAST KEY ACCEPTED BY THE SYSTEM, THEN REQUIRE THAT IT STILL BE PRESSED UPON EXIT FROM THE DECODE ROUTINE, WHICH INCLUDES A DELAY OF APPROXIMATELY 50 MS.
THIS IS AN ATTEMPT TO REDUCE THE NUMBER OF UNINTENTIONAL REPEATED KEYSTROKES. THIS CHANGE INSERTS 'ZAP07' ADDR (IN DOS RESIDENT ROUTINE SYS0) INTO 4016,7 (ROUTINE ADDR IN KEYBOARD CONTROL BLOCK) AND CALLS 03FAH, VICE 03E3H, TO GET THE CURRENT KEY. THIS MIGHT MEAN TROUBLE FOR ANY OF THE FOLLOWING:

1. OTHER USERS WHO CHANGE THIS ADDRESS VALUE IN THE CONTROL BLOCK.
2. ROUTINES WHICH LOAD DIRECTLY OK INDIRECTLY INTO THE DOS AREA. 4000-51FF.

**** NOTE: THIS 'DEBOUNCE ROUTINE' WILL NOT ACTIVATE (LEAVING THE ADDR AT 4016-7 UNCHANGED) IF SHIFT-UP-ARROW IS PRESSED FROM RESET OR POWER-ON UNTIL THE 1ST MSG APPEARS ON THE DISPLAY. WITH THE DEBOUNCE ROUTINE DEACTIVATED, 'JKL' OPERATION IS ALTERED SOMEWHAT. (SEE CHANGE #017).

SYS0, 4E20

***** 017 **

IN CONJUNCTION WITH CHANGE #016, THIS CHANGE ALLOWS THE FULL DISPLAY CONTENTS TO BE SENT AS SIXTEEN 64 CHAR LINES TO THE LINE PRINTER WHENEVER THE 3 KEYS J, K AND L ARE SIMULTANEOUSLY DEPRESSED AND ONE OF THE FOLLOWING EXISTS:

1. 'DEBOUNCE' IS ACTIVATED (SEE CHANGE #017), AND A USER PROGRAM IS TESTING FOR A DEPRESSED KEY. (J, K OR L IS NOT SEEN BY THE CALLING PROGRAM).
2. CLOCK INTERRUPTS ARE ENABLED. IF 'DEBOUNCE' IS NOT ACTIVATED, ONE OR MORE OF J, K OR L MAY BE INPUTTED TO A PROGRAM WAITING ON KEYBOARD INPUT AT THIS TIME.

THE ONLY EDITING DONE BY THIS ROUTINE ENROUTE TO THE PRINTER IS TO CHANGE ALL CHARS WHOSE HEXADECIMAL VALUE >= 80H TO PERIODS. THIS WILL CAUSE GRAPHICS TO PRINT AS PERIODS.

SYS0, 4520 4E20

***** 018 ***

L.2 BASIC DOES NOT FULLY CONVERT BOTH SHIFT AND NON-SHIFT VALUES FOR THE LETTERS A-Z AND THE SYMBOL @ TO ASCII UPPER CASE. (BASIC'S KEY INPUT ROUTINE ACTUALLY CONVERTS THESE CHARS INITIALLY FROM LOWER TO UPPER AND UPPER TO LOWER. THEN THE SOURCE PACK ROUTINE (1BC0) CONVERTS MOST BUT NOT ALL OF THE NON LOWER (ORIGINALLY UPPER) CASE CHARS BACK TO UPPER CASE.) PROPER BASIC PROGRAM OPERATION REQUIRES ALL CHARACTERS IN THE TEXT TO BE UPPER CASE FOR @ AND A-Z. ERRORS IN THIS REGARD ARE NOW CORRECTED WHEN LINE IS ENTERED (OR EDITED). LOWER-CASE TEXT IS RETAINED BY NEWDOS IF SUCH LINES ARE ENTERED AND EDITED BY TRSDOS ONLY.

THIS 'FILTER' MY BE INHIBITED USING SUPERZAP; ON NEWDOS+ DISKETTE TR.3, SEC.5, MOD 6D & 6E FROM 00 52 TO 33 60.

BASIC, 5523

***** 019 **

PROVIDE DOS-BASIC WITH AN OPEN "E" FUNCTION WHICH:

1. OPENS A FILE
2. SETS 'NEXT' INDICATOR = 'EOF', (FCB+5,A,B = FCB+8,C,D)
3. SETS PROCESSING TO CONTINUE AS IF OPEN "O".

THIS ALLOWS THE USER TO ADD TO A SEQUENTIAL FILE.
IF THE FILE WAS CREATED BY OPEN OR WAS NOT YET PREVIOUSLY WRITTEN TO (IE, EOF = 0) THEN OPEN "E" OPERATES EXACTLY AS OPEN "O".

BASIC, 6372, 63B6

***** 020 **

'EOF' FUNCTION WILL MALFUNCTION UNDER THE REMOTE POSSIBILITY OF 'NEXT' > 'EOF' AND THE EOF BYTE AT MISMATCH = 0

BASIC, 61BF, 6200

***** 021 **

'LOC' COMMAND IN DOS-BASIC FAILS DUE TO ERRONEOUS CALL.

BASIC, 6231

***** 022 **

DOS-BASIC'S FILESPEC EDIT ROUTINE ALLOWS ONLY A 22 CHAR FILESPEC WHEREAS SYSTEM DEFINITION ALLOWS FOR 23 (NNNNNNNN/TTT.PPPPPPPP:D)

BASIC, 630C

***** 023 **

'APPEND' DOS COMMAND JUST DOESN'T WORK. HOPEFULLY NOW IT DOES.

SYS6, 573D

***** 024 ***

WHEN DOING LOGICAL TRANSFER (BYTE OR LOGICAL RECORD)(BIT 7,FCB+1 ON), THE OUTPUT SECTOR BUFFER MUST BE DIFFERENT FROM THE INPUT SECTOR BUFFER.

SYS6, 5720

***** 025 ***

'VERIFY 'CHANGES THE WRONG INSTRUCTION IN SYS0, CHANGING THE WRITE-WITH-VERIFY VECTOR RATHER THAN THE WRITE-WITHOUT-VERIFY VECTOR.

SYS6, 550F

***** 026 ***

'ATTRIB' USES THE WRONG ERROR CODE.

SYS6, 5821

***** 027 ***

'ATTRIB' BAD BRANCH

SYS6, 5BA5

***** 028 ***

PROVIDE 'ATTRIB' WITH CAPABILITY UP TURNING OFF THE INVISIBLE ATTRIBUTE.

SYS6, 5C02

***** 029 ***

SYS3 CHANGED TO CAUSE 'KILL' TO ZERO 1ST BYTE OF RELEASED FPDE OR FXDE, RATHER THAN JUST RESETTING THE ENTRY-IN-USE BIT. THIS AGREES WITH 'CLOSE'.

SYS3, 4FAA

***** 030 ***

'CLOSE' IN SYS3 CAUSES A MAJOR SYSTEM DISASTER WHEN IT RELEASES A FXDE BY NOT PRESERVING THE CONTENTS OF REG DE, CONTAINING COUNT+1 OF SECTORS YET TO BE FREED, WHEN FREEING A NO LONGER NEEDED FXDE.
THIS ERROR IS COMPOUNDED BY THE BRANCH AT 4ED9 NOT IMPLICITLY ENDING DEALLOCATION WHEN THE FILE IS KNOWN TO HAVE NO MORE GRANULES ASSIGNED.

THESE ERRORS CAUSE ALL WRITEABLE MAIN MEMORY FROM 3000-42XX TO BE SET = 'FF', WHERE XX IS RELATIVE POSITION WITHIN THE SECTOR OF THE LAST BYTE OF THE FPDE POINTED TO BY THE LAST FXDE RELEASED.
THE CORRESPONDING SECTOR IN THE DIRECTORY IS ALSO FILLED WITH 'FF' TO THAT RELATIVE POINT. FURTHER, AS THIS GOES ON, THE GAT DIRECTORY SECTOR IS MODIFIED TO FREE UP GRANULES AT RANDOM IN TRACKS 00-FF WITH MOST TRACKS BELOW 00H. UNDETECTED, THIS WILL CAUSE GRANULES PREVIOUSLY ALLOCATED TO OTHER FILES TO BE ALLOCATED AGAIN IN SUBSEQUENT FILE ALLOCATIONS.
THIS INCLUDES REALLOCATION OF BOOT/SYS AND DIR/SYS GRANULES, EVENTUALLY CLOBBERING THEM.
ALSO, FILES WHOSE FPDE PRECEDED THE DESTROYED FPDE IN THE DIRECTORY ENTRY SECTOR WILL DISAPPEAR FROM THE SYSTEM AND IF A FILE'S FXDE WAS SO DESTROYED THE USER

SHOULD HAVE HORRENDOUS TROUBLE FROM THAT FILE, CONSIDERED LUCKY IF LUCKY SYSTEM DETECTS AN ERROR.

****** ALMOST AS BAD, HL IS NOT DECREMENTED TO 2ND BYTE OF NEXT LOWER EXTENT NOR IS IT PROTECTED BY THE DIRECTORY SECTOR WRITE CALL AT 4F08. THIS CAUSES THE TWO BYTES (WHATEVER THEY ARE) AT 41FF-4200 TO BE USED AS THE NEXT EXTENT FOR THE FILE, CAUSING A SOMEWHAT RANDOM DEALLOCATION OF GRANULES, USUALLY IN THE RANGE OF TRACKS 00-10H.

YET MORE. IF A NEW FXDE IS ALLOCATED TO THE FILE AND THEN THE DISKETTE IS FOUND TO BE FULL, SYS3 MALFUNCTIONS (AT SOME FUTURE TIME) WHEN CLOSE TRIES TO FREE THE SPACE ASSIGNED TO THAT FXDE, ASSUMING THERE IS SOME WHEN THERE IS NONE.

SYS3, 4ED9, 4FFD, 500C 4F08, 4E92, 4ED1

***** 031 ***

CHANGE TO DOS-BASIC TO ALLOW THE FOLLOWING 'DIRECT' COMMANDS:

1.	'.' (PERIOD)	LIST CURRENT TEXT LINE.
2.	'DOWN-ARROW'	LIST NEXT TEXT LINE. IF NO NEXT TEXT LINE, PERFORMS AS 'SHIFT-DOWN-ARROW'
3.	'UP-ARROW'	LIST TEXT LINE BEFORE CURRENT LINE. IF NONE, PERFORMS AS 'SHIFT-UP-ARROW',
4.	'SHIFT-UP-ARROW'	LIST 1ST TEXT LINE.
5.	'SHIFT-DOWN-ARROW'	LIST LAST LINE 1N TEXT.
6.	',' (COMMA)	'EDIT' CURRENT TEXT LINE.

ONLY 1 SUCH COMMAND PER DIRECT STATEMENT LINE, AND THE COMMAND, TO BE SEEN, MUST BE THE FIRST CHAR OF THE INPUT LINE (NO SEQUENCE NUMBER ALLOWED AND BACKSPACING NOT ALLOWED).

BASIC, 5993

***** 032 ***

IN 'FORMAT', ALLOW THE USER THE OPTION OF CONTINUING THE FORMAT WHEN DISKETTE FOUND TO CONTAIN DATA.

FORMAT, 542C

***** 033 ***

NEED TO POWER UP DRIVE AGAIN IF ADDITIONAL INPUT WAITS PATCHED IN.

FORMAT, 5464

***** 034 ***

'FORMAT' DOES NOT DAMAGE DOS AREAS. THEREFORE A SYSTEM RESET IS NOT NECESSARY. UPON FINAL 'ENTER', EXIT VIA 402DH IF FORMAT COMPLETED AND VIA 4030H IF FORMAT DID NOT FULLY COMPLETE.

FORMAT, 56AA

***** 035 ***

CHANGES TO ALLOW DOS COMMANDS TO BE EXECUTED WITHIN BASIC'S 'CMD' COMMAND.

THE 'CMD' FUNCTION IS USED, PREEMPTING THE SITUATION WHERE THE QUOTE STRING CONTAINS MORE THAN ONE CHAR. IF THE LAST CHAR OF THE QUOTE STRING IS A '$', THEN THE STRING IS CONSIDERED TO BE A CHARACTER STRING VARIABLE NAME, AND THE CONTENTS OF THAT VARIABLE'S STRING IS USED AS THE QUOTE STRING FOR THE 'CMD' FUNCTION AS THE DOS COMMAND.

THIS PATCH USES 444B-4466, AN AREA ASSUMED AVAILABLE IN SYS0.

CMD°XX' WILL CAUSE 'OM' (OUT OF MEMORY) ERROR TO OCCUR IF THERE IS INSUFFICIENT FREE SPACE BETWEEN BOTTOM OF STRING AREA AND TOP OF ARRAY AREA. CURRENTLY, ABOUT 8K OF FREE SPACE IS REQUIRED. BEFORE GOING TO DOS, A CHECKSUM IS COMPUTED OVER THE ENTIRE MAIN MEMORY AREA OF BASIC THAT IS NOT ALLOWED CHANGED BY THE DOS FUNCTION. THIS TAKES ABOUT 2 SECONDS. IF THE DOS FUNCTION RETURNS TO BASIC, THIS CHECKSUM IS COMPUTED AGAIN (ANOTHER 2 SECS). IF THE CHECKSUM IS BAD, BASIC CANNOT CONTINUE AND AN EXIT TO 'DOS READY' IS TAKEN.

THE PROGRAM OR FUNCTION INVOKED BY CMD"XX" MAY RETURN TO BASIC TO CONTINUE THE BASIC PROGRAM'S EXECUTION, IF IT DID NOT USE ANY OF MEMORY BETWEEN 6F00 AND BASIC'S HIGH ADDR (MEN SIZE) AND IT TAKES ONE OF THE FOLLOWING EXIT:

1. 402D SUCCESSFUL COMPLETION EXIT. NEXT BASIC STATEMENT WILL BE EXECUTED.
2. 4030 UNSUCCESSFUL COMPLETION EXIT. COMPLAINT HAS ALREADY BEEN DISPLAYED. IF CMD"D" NOT ACTIVE 'UNPRINTABLE ERROR' (DECIMAL 38) BASIC ERROR WILL BE INVOKED, CAUSING EITHER THE 'ON-ERROR' ROUTINE TO BE INVOKED OR PROGRAM EXECUTION TERMINATED IN ERROR. IF CMD"D" IS ACTIVATED, 4030 EXIT GOES TO DEBUG DISPLAY, FROM WHICH 'RESET' MUST BE ISSUED WHEN DONE WITH DEBUG.
3. 4409 UNSUCCESSFUL COMPLETION EXIT. DEPENDING UPON THE SYSTEM ERROR CODE, A BASIC ERROR CODE WILL BE INVOKED, CAUSING 1 OF THE TWO (CMD"D" INACTIVE) OPTIONS IN THE 4030 EXIT. CMD"E" CAN DISPLAY THE ACTUAL SYSTEM ERROR MSG.

******** CAUTION:** IF THE INVOKED DOS FUNCTION (IE, A SINGLE DRIVE-TWO DISKETTE COPY OR OTHER OBJECT CODE PROGRAMS SUCH AS 'EDTASM') USES BASIC'S MEMORY ABOVE 6F00, EXITING THROUGH ONE OF THE ABOVE THREE ADDRESSES WILL CAUSE UNPREDICTABLE RESULTS AS BASIC'S CODE IS CLOBBERED. WITH LUCK, RESET WILL TAKE PLACE.

BASIC, 56CD
SYS0, 444B-4466

***** 036 ***

CHANGES TO SPEED UP 'BASIC'S 'LOAD' AND 'SAVE' OPERATIONS BY USING SECTOR I/O TO DOS SYSTEM RATHER THAN BYTE I/O.
FOR LARGE BASIC PROGRAMS, LOAD AND SAVE RUN AT APPROX. 33% AND 50% OF PREVIOUS TIMES RESPECTIVELY.

BASIC, 5FB6, 6055

***** 037 **

CHANCES TO FORMAT TO ALLOW IT TO EXECUTE WITH PARAMETERS SPECIFIED ALONG WITH THE FORMAT COMMAND.

1. FORMAT D,NNNNNNNN,MM/DD/YY,PPPPPPPP
2. FORMAT D,NNNNNNNN,MM/DD/YY,PPPPPPPP,Y

WHERE

D = RELATIVE NUMBER OF DRIVE TO BE FORMATTED
NNNNNNNN = 1-8 CHAR NAME FOR DISKETTE
MM/DD/YY = CREATION DATE
PPPPPPPP = 1-8 CHAR PASSWORD
Y PARAM IS USED IF OK THAT DISKETTE BEING FORMATTED CONTAINS DATA.

IF 'D' IS DRIVE 0 (THE SYSTEM DRIVE), FORMAT WILL ASK FOR THE PROPER FORMAT/DESTINATION OR SOURCE DISKETTE TO BE MOUNTED, AND UPON EXIT WILL ASK FOR A SYSTEM DISKETTE TO BE MOUNTED BACK ONTO DRIVE 0.

'FORMAT' IS ALSO MODIFIED SUCH THAT IF COMMAND PARAMS ARE SUPPLIED WITH THE COMMAND, THE COMMAND CAN BE RETRIED AFTER USER HAS CORRECTED THE PROBLEM, PROVIDED THE COMMAND PARAMS THEMSELVES ARE NOT IN ERROR.

FORMAT, 522E, 5278, 52A0, 52AE, 52B6; 52BE, 52CF, 5341

***** 038 **

FORMAT IS ALTERED TO USE 5D00 AS BASE OF FULL TRACK BUFFERS, VICE 5F00. THIS DECREASES FORMAT EXECUTION AREA SIZE TO REDUCE SIZE SAVED BY BASIC WHEN EXECUTING DOS COMMANDS.

FORMAT, 5499, 5526, 5551

***** 039 **

'BASIC IS MODIFIED SUCH THAT ITS ACTIVATION SEQUENCE IS ONE OF THE FOLLOWING:

1. BASIC
2. BASIC *
3. BASIC N
4. BASIC M
5. BASIC CMD
6. BASIC N,M,CMD
7. BASIC M,N,CMD
8. BASIC N,M
9. BASIC M,N
10. BASIC N,CMD
11. BASIC M,CMD

WHERE:

* MEANS THE USER WANTS BASIC TO REINSTITUTE THE PROGRAM IN THE TEXT BUFFERS USING THE SAME VALUES FOR M AND N AS APPEAR TO EXIST IN MAIN MEMORY. THIS ALLOWS THE USER TO RECOVER FROM AN UNWANTED 'RESET' OR TO GET BACK TO THE SAME PROGRAM AFTER A CMD"S". IF BASIC IS ABLE TO ACCOMPLISH THIS, IT FORCES 'LIST' AS ITS 1ST COMMAND.

N = # OF I/O AREAS. DEFAULT = 3. MAX = 15.

M = MEMORY SIZE. DEFAULT IS ALL THAT THE COMPUTER HAS.

CMD = ANY BASIC DIRECT COMMAND. THIS COMMAND WILL BE EXECUTED AS SOON AS INITIALIZATION IS COMPLETED.

ANY ERROR ENCOUNTERED DURING INITIALIZATION CAUSES A RETURN TO DOS. THE OLD BASIC STARTUP PROCEDURE IS HEREBY SUPERSEDED.

BASIC, 5200, 5BB0, 5BF0, 5C35, 5C54, 5CA4, 5C65, 5C88

***** 040 **

'COPY' IS ALTERED TO HANDLE THE FORMATS:

1. COPY FILESPEC1 TO FILESPEC2 (SAME AS OLD COPY)
2. COPY $FILESPEC1 TO FILESPEC2
3. COPY :D FILESPEC1 TO FILESPEC2
4. COPY :D $FILESPEC1 TO FILESPEC2
5. COPY :D TO :E MM/DD/YY

WHERE:

1. D AND E ARE DRIVE #'S AND MAY BE EQUAL.
2. FILESPEC1 IS THE 'SOURCE' FILE'S FILESPEC.
3. FILESPEC2 IS THE 'DESTINATION' FILE'S FILESPEC.
4. $ MEANS THAT EITHER THE SOURCE OR THE DESTINATION FILE OK BOTH ARE ON A DISKETTE CONTAINING AN ALIEN SYSTEM (IE, NOT THIS SYSTEM) IF D IS SPECIFIED AND IS NOT = 0 THEN $ SHOULD NOT BE SPEC'ED AS THE NECESSARY COMPATIBLE SYSTEM ROUTINES, SYS2 AND SYS3 WILL BE TAKEN FROM DRIVE 0, WHICH IS ASSUMED TO CONTAIN 'THIS' SYSTEM.
5. FORMATS 3 AND 4 INDICATE THAT SOURCE AND DESTINATION FILES ARE ON DIFFERENT DISKETTES, BUT THAT ONLY THE SPECIFIED DRIVE WILL BE USED TO CONTAIN THOSE DISKETTES.
6. FORMAT 5 PERFORMS A BACKUP FUNCTION. BEFORE DATA IS WRITTEN ON THE DESTINATION DISKETTE, IT IS FORMATTED TO ALL DATA SECTORS (NO BOOT OR DIRECTORY). THE ENTIRE CONTENTS OF THE SOURCE DISKETTE IS THEN COPIED ONTO THE DESTINATION DISKETTE (INCLUDING BOOT AND DIRECTORY), AND THE 'MM/DD/YY' VALUE INSERTED AS THE CREATION DATE FOR THE DESTINATION DISKETTE.
7. WHEN TWO DISKETTES, BUT ONLY 1 DRIVE, ARE SPECIFIED FOR A COPY, ALL OF MEMORY ABOVE COPY'S CODE IS USED AS BUFFERS. ALL OTHER CASES USE A ONE TRACK BUFFER, THUS ALLOWING RETURN TO BASIC, IF COPY WAS INVOKED VIA BASIC'S CMD"XX" STATEMENT.
8. WHENEVER A SPACE IS USED AS A SEPARATOR, ONE AND ONLY ONE IS ALLOWED.

WHEN THE LEADING PORTIONS OF FILESPEC2 ARE THE SAME AS THAT OF FILESPEC1, IT MAY BE LEFT OFF FILESPEC2 WITH FILESPEC2 STARTING WITH ONE OF / OR . OR : TO INDICATE THE DIFFERING PORTION.
EXAMPLE:

```
      COPY ABCD/TTT:0 TO ABCD/TTT:1
```

MAY HE WRITTEN AS

```
      COPY ABCD/TTT:0 TO :1
```

'COPY' IS NO LONGER EXECUTED AS A FUNCTION WITHIN SYS6. SYS1 IS MODIFIED TO ROUTE 'COPY' TO COPY/CMD MODULE. SYS6 IS MODIFIED TO NOT USE ITS 'COPY' FUNCTION. DUE TO ATTEMPTS TO RESTRICT COPY/CMD TO 1 GRANULE, THE FORMAT PORTION OF COPY HAS BEEN PLACED IN SYS3, INVOKED BY THE 70 CODE.

ALSO, OLD 'COPY' MALFUNCTIONED IF LAST SECTOR BEFORE EOF SECTOR IS NOT FULL (IE, FPDE+3 AND FCB+8 <> 0) BECAUSE IT SECTOR READ/WRITES RATHER THAN SINGLE BYTE READ/WRITES. THIS CHANCE CORRECTS THIS.

COPY/CMD
SYS1, 4EDD
SYS6, 5216
SYS3, ZAP15, FORMAT

***** 041 ***

PROGRAM LOADER IN SYS0 IS ALTERED TO USE 4200-42FF, VICE 4D00-4DFF AS IT'S I/O BUFFER, THUS ALLOWING SYSTEM PROGRAMS TO LOAD INTO 4D00-4DFF, IF DESIRABLE. A CHECK IS MADE IN THE ACTUAL I/O CODE TO DISABLE REMEMBRANCE OF THAT SYSTEM PROGRAM BEING IN MAIN MEMORY IF ITS ENTRY POINT WAS WITHIN 4D00-4DFF AND A READ/WRITE SECTOR TAKES PLACE ON THAT AREA.

SYS0, 4C18, 44A4, 4C39

***** 042 ***

IF A SYSTEM NODULE BEING LOADED USING THE 16 BYTE FCB (HALF SIZE) AT 44A0, AN 'RECORD OUT OF RANGE' (1DH) ERROR IS CALLED IF FILE HAS MORE THAN ONE EXTENT.

SYS0, 4912

***** 043 **

IF THE SYSTEM MODULE 'CALL' ROUTINE FINDS THE MODULE'S DIRECTORY ENTRY INACTIVE, THEN ONE OF THE FOLLOWING:

1. IF SYS4 IS AN ACTIVE MODULE IN THE SYSTEMS THEN 'FILE NOT IN DIRECTORY' ERROR WILL BE DISPLAYED VIA JUMP TO SYS4 AT 4409H (IF CMD"XX", BASIC WILL INTERCEPT).
2. IF THE JUMP TO SYS4 VIA 4409 FINDS SYS4 NOT IN THE SYSTEM, THEN 'RESET' WILL BE INVOKED.

SYS0, 4BD4

***** 044 **

TO REDUCE THE RESIDENT SIZE OF BASIC (DUE: TO ADDED CODE) THE FOLLOWING STEPS HAVE BEEN TAKEN:

1. SYS13/SYS MODULE CREATED TO CONTAIN DOS BASIC'S ERROR DISPLAY CODE AND TABLE,
2. 5200-5555: ENTRY TO 5200 IS CHANGED TO ZAP18. ERROR MESSAGES AND ERROR DISPLAY CODE FROM 57B2-57E0 ARE MOVED TO SYS13. THE JUMP VECTORS (54C3-5555) ARE MOVED TO 4D00 AS THEY ARE NEEDED ONLY DURING INITIALIZATION.
3. 5782-5623: THE ERROR DISPLAY CODE (57B2-57E0) IS MOVED TO SYS13. CODE (57E1-573F) IS DELETED AND CMD"X" DISABLED. THE AUTHORED/COPYRIGHTED MSG IS APPENDED, DISPLAYED DURING INITIALIZATION AND THEN ITS STORAGE REUSED.
4. 5BA4-5DC7: INITIALIZATION CODE GREATLY MODIFIED. COMMAND BUFFER SHIFTED TO 5BAD-5C9D TO OVERLAP INITIALIZATION CODE, WHICH IS NOT NEEDED AFTER INITIALIZATION. THE ROUTINE AT 5C8B-5C90 NOW AT 5BA4-5BA9.
5. 5A15-5B13: ROUTINE AT 5A5B-5A66 MOVED TO 5A15-5A20. HAVE NO IDEA WHAT THE REST OF THE AREA WAS USED FOR; SO WILL USE 1T FOR APPENDAGE CODE.
6. IF NECESSARY, MORE APPENDAGE CODE CAN BE ADDED AT 6431.

***** 045 ***

THROUGH NEW SYSTEM MODULE 'SYS12/SYS' THE FOLLOWING BASIC COMMAND IS ALLOWED!

1. REF* DISPLAY FULL REFERENCE LIST.
2. REF$ PRINT FULL REFERENCE LIST ON LINE PRINTER.
3. REFNN DISPLAY ALL REFERENCES TO THE VARIABLE(S) NAMED NN. IF NN IS ONLY 1 CHAR, BLANK IS ASSUMED FOR THE SECOND. NN MAY NOT BE MORE THAN 2 CHARS AND MUST NOT HAVE A TYPE SUFFIX.
4. REFSSSSS DISPLAY ALL REFERENCES TO THE INTEGER SSSSS WHERE SSSSS IS A 1-5 DECIMAL DIGIT # BETWEEN 0 AND 99999. HEXADECIMAL OR OCTAL REFERENCES WITHIN THE TEXT ARE NOT LISTED.
5. REF*NN
6. REF$NN
7. REF*SSSSS
8. REF$SSSSS
9. REF DISPLAY THE NEXT TEXT LINE CONTAINING AT LEAST ONE REFERENCE TO THE VARIABLE OR NUMBER SPECIFIED BY THE LAST REFNN OR REFSSSSS COMMAND EXECUTED. IF NO MORE REFERENCING TEXT LINES, 'TEXT END' WILL BE DISPLAYED.
   IF 'REF' ENTERED AGAIN, THE FIRST REFERENCING TEXT LINE WILL BE LISTED.
   REMEMBRANCE OF THE LAST VARIABLE NAME, NUMBER OR POSITION WITHIN THE TEXT IS USUALLY (BUT NOT ALWAYS) LOST WHEN ANY COMMAND INVOLVING DOS IS EXECUTED.

PRESS BREAK TO PAUSE, ENTER TO CONTINUE.

FORMATS 5-8 ARE THE SAME AS 1 AND 2, EXCEPT LISTING/PRINTING STARTS WITH THE SPECIFIED VARIABLE NAME OR DECIMAL NUMBER, IF IT EXISTS, OR THE NEXT HIGHER EXISTING NAME OR NUMBER, IF NOT.

IF 'SYS12/SYS' NOT IN THE SYSTEM WHEN COMMAND 'REF' EXECUTED, BASIC ERROR 'UNPRINTABLE ERROR' (CODE 38 DECIMAL) WILL BE INVOKED. CMD"E" WILL DISPLAY 'FILE NOT IN DIRECTORY'

***** 046 ***

NEW SYSTEM MODULE 'SYS11/SYS' ALLOWS THE FOLLOWING BASIC COMMAND:

1. RENUM U SEARCHES TEXT FOR UNDEFINED SEQUENCE NUMBERS AND FOR SOME ERRORS ASSOCIATES WITH BASIC STATEMENTS THAT USE SEQUENCE NUMBERS.
   ERRORS ARE INDICATED:
   SSSSS/U = THERE IS NO TEXT LINE SSSSS.
   SSSSS/X = TEXT LINE SSSSS HAS SYNTAX ERROR.
   SSSSS/S = TEXT LINE SSSSS HAS A BAD SEQUENCE #.

   TEXT IS NOT ALTERED IN ANY WAY.

2. RENUM SSSSS,IIIII,PPFPP,QQQQQ
   CAUSE ALL TEXT LINES WHOSE SEQ #'S ARE >= PPPPP AND <= QQQQQ TO BE ASSIGNED NEW SEQ #'S. SSSSS IS THE FIRST NEW SEQ # ASSIGNED WITH SUBSEQUENT #'S GENERATED BY ADDING IIIII TO THE SEQ # OF THE PREVIOUS TEXT LINE. SSSSS AND IIIII MUST BE IN THE RANGE 1 - 65529 AND HAVE DEFAULT VALUE 10. PPPPP MUST BE IN THE RANGE 1 - 65529, HAS DEFAULT VALUE 0 AND MUST BE <= SSSSS. QQQQQ MUST BE IN THE RANGE 1 - 65529, >= SSSSS, AND HAS DEFAULT VALUE 65529.

COMMANDS USING THE DEFAULTS ARE:

```
RENUM ,
RENUM SSSSS
RENUM ,IIIII
RENUM ,,PPPPP
RENUM ,,,QQQQQ
RENUM SSSSS,IIIII
RENUM SSSSS,,PPPPP
RENUM ,IIIII,PPPPP
RENUM SSSSS,,,QQQQQ
RENUM ,IIIII,,QQQQQ
RENUM ,,PPPPP,QQQQQ
RENUM SSSSS,IIIII,PPPPP
RENUM SSSSS,IIIII,,QQQQQ
RENUM SSSSS,,PPPPP,QQQQQ
RENUM ,IIIII,PPPPP,QQQQQ
```

IF ANY ERROR ENCOUNTERED BEFORE TEXT IS ALTERED, THE COMMAND REVERTS TO 'RENUM U' AND DISPLAYS ALL ERRORS IT CAN FIND. IF AN ERROR ENCOUNTERED AFTER TEXT ALTERATION BEGINS, 'FATAL ERROR. TEXT NOW BAD' IS DISPLAYED AND A 4030 EXIT TAKEN, GOING EITHER TO 'DOS READY' OK TO DEBUG (IF CMD"D" ACTIVE). THE BASIC TEXT MUST NOT BE RECLAIMED (DON'T USE BASIC *).

IF 'SYS11/SYS' NOT IN THE SYSTEM WHEN COMMAND 'RENUM' IS EXECUTED, BASIC ERROR 'UNPRINTABLE ERROR' (CODE 30 DECIMAL) WILL BE INVOKED, WITH CMD"E" DISPLAYING 'FILE NOT IN DIRECTORY'.

***** 047 ***

BASIC IS ALTERED TO ALLOW L, E OR D IN PLACE OF LIST, EDIT OR DELETE RESPECTIVELY WHEN EACH IS ALL OF THE FOLLOWING:

1. 1ST CHAR OF INPUT LINE.
2. FOLLOWED BY EITHER A PERIOD OR A DECIMAL DIGIT.
3. THE INPUT LINE DOES NOT CONTAIN AN :.

***** 048 ***

PROGRAM DISKDUMP/BAS HAS BEEN UPGRADED TO ALLOW:

1. OPTION OF SENDING DUMP TO PRINTER OR DISPLAY.
2. OPTION OF DUMPING BY DISK RELATIVE SECTOR WITHIN FILE OR LOGICAL RECORD # WITHIN FILE (RECORDS END WITH CARRIAGE RETURN. (ASC CODE 13 DECIMAL). EACH OCCURRENCE OF THIS CHAR IS TREATED AS AN EOR REGARDLESS OF OTHER SPECIAL MEANINGS (IE, WITHIN A QUOTE STRING OR PRECEDED BY A DOWN-ARROW).
3. A DUMP PAUSE COMMAND.

***** 049 ***

PROGRAM EDTASM/CMD HAS BEEN UPGRADED TO:

1. READ SOURCE FROM DISK AS WELL AS CASSETTE.
2. WRITE SOURCE AND/OR OBJECT TO DISK AS WELL AS CASSETTE. DISK FILES ARE VALIDITY READ AFTER ALL SECTORS WRITTEN.
3. ALLOW 'DOWN-ARROW' SCROLLING TO DISPLAY UP TO 15 TEXT LINES.
4. PREVENT THE CONFUSING PRINTER OUTPUT ASSOCIATED WITH 'DEFM'. ONLY THE 1ST BYTE OF ASSOCIATED OBJECT CODE IS LISTED.
5. LIST SYMBOLS IN ALPHABETICAL ORDER WITH REFERENCE LIST.
6. ACCEPTS AND CONVERTS LOWER CASE ALPHA TO UPPER.

SUPPLEMENTAL INSTRUCTIONS TO THE EDITOR ASSEMBLER.

1. TO LOAD A SOURCE MODULE INTO THE TEXT BUFFER.
    1. L D=NNNNNNNN/TTT.PPPPPPPP:D IF SOURCE FROM DISK.
    2. L T=NNNNNN IF SOURCE FROM CASSETTE.

   IF THE TEXT BUFFER ALREADY CONTAINS TEST, THE QUERY "TEXT IN BUFFER. ARE YOU CONCATENATING???" APPEARS. REPLY 'N' IF NOT, CAUSING BUFFER TO BE SET EMPTY BEFORE NEW MODULE LOAD. REPLY 'Y' IF YOU ARE, CAUSING THE NEW SOURCE TO BE APPENDED ONTO THE END OF THE OLD. NO CONCERN IS SHOWN FOR OVERLAPPING SEQ #, THEREFORE YOU SHOULD EXECUTE AN 'N' EDTASM COMMAND UPON COMPLETION OF THE LOAD TO ASSURE A VALID SET OF ASCENDING SEQUENCE NUMBERS.

2. TO STORE A SOURCE MODULE
    1. W D=NNNNNNNN/TTT.PPPPPPPP:D IF SOURCE GOING TO DISK.
    2. W T=NNNNNN IF SOURCE GOING TO CASSETTE.

3. FOR A COMMANDS WITH 'NO' OPTION NOT SPECIFIED, RESPOND TO THE QUERY "OBJECT FILE TO DISK OR TAPE? REPLY D OR T?":
    1. T IF OBJECT GOING TO CASSETTE. NAME WILL COME FROM THE A COMMAND.
    2. D IF OBJECT GOING TO DISK. RESPOND TO THE QUERY "OBJECT FILESPEC?" WITH THE NNNNNNNN/TTT.PPPPPPPP:D FILESPEC OF THE OBJECT MODULE. THE FILE WILL BE OPENED IMMEDIATELY, BUT NOT WRITTEN UNTIL END OF ASSEMBLY LISTING. THE NAME IN THE A COMMAND IS IGNORED.

4. WHEN AN OUTPUT SOURCE OR OBJECT FILE IS OPENED, ONE OF THE FOLLOWING IS DISPLAYED:
    1. "FILE ALREADY EXISTS. USE IT????". REPLY 'Y' IF THIS IS YOUR INTENTION. OTHERWISE REPLY 'BREAK' TO TERMINATE THE W OR A COMMAND.
    2. "*************** FILE NON-EXISTENT. REPLY "C" TO CREATE IT". REPLY 'C' IF THIS IS YOUR INTENTION. OTHERWISE REPLY 'BREAK' TO TERMINATE THE W OR A COMMAND.

5. DUE TO AN ERROR IN THE ORIGINAL DOS, EDTASM RUNS WITH INTERRUPTS DISABLED (EXCEPT WHEN RE-ENABLED BY DISK I/O) IN ORDER THAT USE OF 'BREAK' WILL FUNCTION PROPERLY.

6. THIS EDTASM CAN EXECUTE IN A REGULAR TRSDOS ENVIRONMENT.

7. THIS EDTASM REQUIRES A TRSDOS-LIKE OPERATING SYSTEM, AND USES LEVEL II KEYBOARD, DISPLAY AND PRINTER ROUTINES AND CONTROL BLOCKS. USERS ALTERING THE SYSTEM BEWARE!!!!

***** 050 ***

BASIC1/CMD IS A MODIFIED VERSION OF THE ORIGINAL LEVEL I BASIC. THIS MODULE LOADS FROM DISK INTO 6E00-7FFFH, AND CAN FUNCTION AS LEVEL I FOR LEVEL I BASIC PROGRAMS THAT EXECUTE IN A REGULAR LEVEL I COMPUTER WHERE "MEM SIZE" IS SPECIFIED AS 28160 (6E00H) OR LESS. HOWEVER, DUE TO THE WIDE VARIETY OF LEVEL I BASIC PROGRAMS, APPARAT DUES NOT GUARANTEE THE OPERATION OF ANY OF THEM VIA BASIC1/CMD.

THE FOLLOWING COMMANDS ARE ADDED TO LEVEL 1:

1. CSAVE* THIS CRUSES THE PROGRAM TEXT TO BE MOVED INTO MAIN MEMORY AT 8000H AND UP, RATHER THAN SENT TO THE CASSETTE. 'RESET' AND THEN DOS COMMAND 'LV1DSKSL' WITH THE 'S' OPTION WILL STORE THE LEVEL I PROGRAM LOCATED IN THE 8000-UP REGION ONTO DISK.

2. CLOAD* FIRST, EXECUTE DOS COMMAND 'LV1DSKSL' WITH 'L' OPTION. THE SPECIFIED LEVEL I PROGRAM WILL BE LOADED FROM DISK INTO MAIN MEMORY BEGINNING AT 8000H. AT. A1 THE CONCLUSION OF THIS LOAD, DOS COMMAND 'LEVEL1' WILL AUTOMATICALLY BE EXECUTED, THUS BRINGING UP LEVEL I. ISSUING LEVEL I COMMAND 'CLOAD*' WILL CAUSE THE LEVEL I TEXT BUFFER TO BE LOADED FROM MAIN MEMORY AT 8000H, RATHER THAN FROM CASSETTE.

3. CSAVE$$$ A JUMP IS TAKEN TO LOCATION 0 TO CAUSE A POWER-ON LEVEL II START UP. BUT WITHOUT LOSING THE CONTENTS OF MEMORY. THIS IS INTENDED AS AN AID TO THE NON-DISK USER WHERE 'RESET' IN A SYSTEM WITHOUT THE EXPANSION INTERFACE LEAVES THE USER STILL IN LEVEL I. RESPONDING MEM SIZE = 28160 OR LESS WILL PRESERVE LEVEL I IN MEMORY FOR SUBSEQUENT STARTUP VIA:
   1. 'SYSTEM'
   2. '/28160'

***** 051 ***

'LV1DSKSL' THIS OBJECT MODULE PROVIDES AN INTERFACE BETWEEN LEVEL-I EXECUTING-IN-LEVEL-II AND THE LEVEL I BASIC PROGRAMS STORED ON DISK.

TO THE QUERY "REPLY "S" FOR SAVES, "L" FOR LOAD.", REPLY "S" IF A LEVEL I BASIC PROGRAM IS IN MAIN MEMORY AT 8000H AS A RESULT OF A 'CSAVE*' LEVEL I COMMAND SUCCESSFUL EXECUTION, AND IT IS TO BE STORED ON DISK. REPLY "L" IF AN EXISTING LEVEL I PROGRAM ON DISK IS TO BE LOADED INTO MAIN MEMORY AT 8000H, LEVEL I ACTIVATED VIA DOS COMMAND 'LEVEL1', AND THE PROGRAM TO LOADED INTO LEVEL I'S TEXT BUFFER BY A SUBSEQUENT 'CLOAD*' ISSUED BY THE USER.

RESPOND TO THE QUERY "FILE-SPEC" WITH THE FILESPEC THE LEVEL I BASIC TEXT MODULE HAS OR IS TO HAVE.

IF "BAD TEXT PREFIX" DISPLAYED, ONE OF THE FOLLOWING EXISTS:
1. TEXT FROM DISK IS NOT A LEVEL I BASIC TEXT MODULE.
2. MAIN MEMORY IS NOT GOOD.
3. USER FORGOT TO EXECUTE 'CSAVE*'.

PROGRAM: 'DISASSEM' VERSION 1.1

THIS DISASSEMBLER DISASSEMBLES Z-80 CPU CODE, AND IS DESIGNED TO RUN IN THE TRS-80 DOS SYSTEM USING AS SOURCE EITHER A STANDARD TRS-80 LOAD MODULE ON DISK OR A PREVIOUSLY LOADED OBJECT MODULE IN MAIN MEMORY.
DISASSEMBLER WRITTEN IN Z-80 CODE.

OPERATION:

ENTER DOS COMMAND 'DISASSEM'

RESPONSES TO QUERY 'OBJECT FROM MAIN MEMORY OR DISK 4
1. NULL SAME AS 'D'
2. 'D' OBJECT IS A DISK LOAD MODULE
   1. RESPONSE TO QUERY 'FILESPEC'?
      1. FILESPEC IN 'NNNNNNNN/TTT.PPPPPPPP:D' FORMAT
   2. RESPONSE TO QUERY 'OFFSET OBJECT VIRTUAL ADDRESSES BY (HEX)':
      1. NULL = 0
      2. 'NNNN' 1 TO 4 HEX DIGIT VALUE WHICH WHEN ADDED TO LOAD ADDRESSES WITHIN THE LOAD MODULE WILL GIVE PROPER ADDRESS WHERE THE CURRENT INST BEING DISASSEMBLED WOULD BE DURING NORMAL EXECUTION OF THAT CODE. THIS PARAMETER IS NEEDED WHEN AN OBJECT MODULE LOADS IN ONE PLACE IN MAIN MEMORY, BUT ACTUALLY EXECUTES FROM ANOTHER. WRAP-AROUND IS ALLOWED. EXAMPLE: IF THE OBJECT MODULE LOADS INTO C000-FFFF BUT IS TO EXECUTE IN 7000-AFFF, APPLYING AN OFFSET OF 5000 WILL CAUSE THE DISASSEMBLER TO DISASSEMBLE AS IF THE LOAD WAS ACTUALLY DONE TO 7000-AFFF.
   3. RESPONSE TO QUERY 'VIRTUAL RESTART LOCATION (HEX)'
      1. NULL NONE
      2. 'NNNN' 1 TO 4 HEX DIGIT VIRTUAL RESTART LOCATION. ALLOWS RESTART OF A LARGE DISASSEMBLY WITHIN THE INSTRUCTION PRINT PORTION OF THE LISTING. 'NNNN' IS THE LISTED LOC OF ANY INSTRUCTION OF THE DISASSEMBLY, USUALLY CHOSEN AS THE LOCATION VALUE PRINTED FOR THE FIRST INST ON THE PAGE WHERE PRINTING WAS INTERRUPTED.

3. 'M' OBJECT IS IN MAIN MEMORY
   1. RESPONSE TO QUERY 'OBJECT VIRTUAL BASE ADDRESS (HEX)'?
      1. 'NNNN' 1 TO 4 HEX DIGIT LOCATION VALUE WHERE THE OBJECT CODE IS CONSIDERED TO EXECUTE FROM. IN THE LISTING, THIS VALUE WILL BE THE FIRST INSTRUCTION PRINTED'S LOCATION VALUE.
   2. RESPONSE TO QUERY 'OBJECT REAL BASE ADDRESS (HEX)'?
      1. NULL REAL LOCATION SAME AS VIRTUAL
      2. 'NNNN' 1 TO 4 HEX DIGIT MEMORY LOCATION WHERE THE DISASSEMBLER WILL ACTUALLY FIND THE OBJECT CODE.

RESPONSES TO QUERY 'OUTPUT TO PRINTER'?
1. 'Y' - OUTPUT IS TO LINE PRINTER. ALL LINES <= 64 CHARS.
   1. RESPONSE TO REQUEST 'REPLY 'ENTER' WHEN PRINTER AT 'TOP OF PAGE'?
      1. NULL SET PRINTER EXACTLY AT TOP OF PAGE. THIS PROGRAM ASSUMES 66 LINES/PAGE AND LEAVES THE TOP 3 AND BOTTOM 3 LINES BLANK ON EVERY PAGE.
   2. IF OBJECT FROM MAIN MEMORY. RESPOND TO QUERY 'BYTE COUNT (HEX)'?
      1. 'NNNN' 1 TG 4 HEX DIGIT COUNT OF BYTES TO DISASSEMBLE.
      2. NULL SAME AS 'N'
      3. 'N' LISTINGS WILL GO TO DISPLAY. IF OBJECT FROM DISK, THE LOCATION TABLE WILL NOT BE LISTED.

RESPONSES TO QUERY 'NORMAL DISPLAY PAUSES'?

1. NULL SAME AS 'Y'
2. 'Y' DURING DISASSEMBLED INSTRUCTION LISTINGS, THE PROGRAM WILL PAUSE AT THE END OF EACH DISPLAY PAGE. TO CONTINUE ON TO THE NEXT PAGE, HIT 'ENTER'. A REPLY OF 'X' WILL TERMINATE THE DISASSEMBLY.
3. 'N' THE DISASSEMBLY WILL DISPLAY AS FAST AS THE PROGRAM CAN HOBBLE. TO PAUSE, HIT 'F'. TO CONTINUE AFTER PAUSE, HIT 'ENTER'.

RESPONSES TO QUERY 'ANY OPTIONS'?. (OBJECT FROM DISK, ONLY)

1. NULL NO MORE OPTIONS
2. 'NIP' DO NOT PRINT OR DISPLAY THE DISASSEMBLED INSTRUCTIONS. THIS DOES NOT AFFECT THE DISPLAY OF THE LOCATION-CROSS-REFERENCE PASS.
3. 'RTD' LOCATION REFERENCE TABLE IS TO BE STORED ON DISK. AFTER THE LOCATION-CROSS-REFERENCE PASS, THE PROGRAM WILL QUERY 'REFERENCE TABLE FILESPEC'? RESPOND WITH THE FILESPEC IN NNNNNNNN/TTT.PPPPPPPP:D FORMAT IDENTIFYING THE FILE TO CONTAIN THE REFERENCE TABLE. REFERENCE TABLE FILES ON DISK CAN BE USED (BY USER-CREATED PROGRAM) TO MERGE THE REFERENCE TABLES OF TWO-OR MORE PROGRAMS, SEE BELOW FOR FILE FORMAT,
4. 'REA' ENABLE LISTING OF ALL TYPES OF REFERENCES; IS THE DEFAULT CONDITION.
5. 'RE&' ENABLE LISTING OF THE SPECIFIED REFERENCE TYPE WHERE & IS ONE OF L, P, R, S, T, U, V, W OR X. REFERENCE TYPES ARE DEFINED AT THE BEGINNING OF EACH LOCATION TABLE LISTING.
6. 'RIA' DISABLE LIST OF ALL TYPES OF REFERENCES
7. 'RI&' DISABLE LISTING OF THE SPECIFIED REFERENCE TYPE WHERE & IS ONE OF L, P, R, S, T, U, V, W OR X.

THE DISASSEMBLER OPERATES IN THESE PHASES.

1. (OBJECT FROM DISK) BUILD LOCATION REFERENCE TABLE. IF INSUFFICIENT MEMORY AVAILABLE, 'INSUFFICIENT MEMORY' WILL BE PRINTED AND THE DISASSEMBLY TERMINATED.
   DURING THIS PHASE, THE OBJECT MODULE WILL BE DISASSEMBLED TO OBTAIN REFERENCE INFORMATION AND WILL DISPLAY THIS PASS WITHOUT PAUSE.
2. WRITE REFERENCE TABLE FILE. (RTD OPTION ONLY)
3. LIST DISASSEMBLED INSTRUCTIONS. IF SPEC'ED, DISPLAY PAUSES WILL OCCUR.
4. PRINT LOCATION REFERENCE TABLE. (OBJECT FROM DISK ONLY AND PRINTER USE SPEC'ED)

IF THE DOS OPERATING SYSTEM RETURNS AN ERROR CODE, '&& DISK ERROR' MSG IS DISPLAYED AND THE DISASSEMBLY TERMINATED. && IS A TWO HEX DIGIT ERROR CODE.
SOME OF THESE CODES ARE:

1C OR 1D	EOF (OBJECT NOT A LOAD MODULE ?????)
25	OBJECT IS READ-PROTECTED
1B	NO MORE SPACE ON DISKETTE
19	OBJECT IS PASSWORD PROTECTED
18	OBJECT FILE DOES NOT EXIST

'DISK OBJECT FILE FORMAT NOT AS EXPECTED' MSG WELL BE DISPLAYED IF THE DISASSEMBLER FINDS SOMETHING WRONG WITH THE OBJECT MODULE.

FILE INSTRUCTION DISPLAYING OR PRINTING IS IN PROGRESS, HOLDING DOWN THE 'X' OR THE 'P' KEY WILL CAUSE DISASSEMBLY TERMINATION OR PAUSE RESPECTIVELY.
HIT THE 'ENTER' KEY TO CONTINUE AFTER PAUSE.

SUFFIXED TO EACH REFERENCING LOCATION VALUE IS A REFERENCE TYPE CODE (DEFINED AT TOP OF EACH REFERENCE LISTING) GIVING THE TYPE OF Z-80 INSTRUCTION MAKING THE REFERENCE.

ON THE DISASSEMBLED INSTRUCTION PRINT LINES:

1. COLUMN 1 INDICATES THE NUMBER OF REFERENCES TO BYTES OF THE INSTRUCTION. VALUE IS HEXADECIMAL WITH BLANK = 0 AND F MEANS 15 OR MORE REFERENCES.
2. COLUMN 2 INDICATES WHICH BYTES OF THE INSTRUCTION HAVE BEEN REFERENCED. IF BLANK AND COLUMN 1 NON-BLANK, THEN ONLY THE INST'S 1ST BYTE REFERENCED OTHERWISE THE HEX DIGIT REPRESENTS A 4-BIT BINARY MASK OF WHICH BYTES, FROM THE LEFT, ARE REFERENCED.

PROGRAM WARRANTY IS LIMITED TO THE PROVISION ON EITHER DISK OR TAPE (SELLER'S DISCRETION) AN OPERATIONAL LOAD MODULE TO RUN UNDER TRS-80 DOS 1.1
THERE IS NO WARRANTY FOR CONTINUED OPERATION UNDER SUBSEQUENT DOS RELEASES.

THIS PROGRAM LOADS INTO 8200, THEN SHIFTS ITSELF TO 5200 FOR EXECUTION.
ENTRY POINT = 9B00H.

TO LOAD THIS MODULE FROM TAPE TO DISK, USE TRS-80'S 'TAPEDISK' PROGRAM.
NAME ON TAPE IS 'DISASM'
NAME ON DISK IS 'DISASSEM/CMD'

FORMAT OF REFERENCE TABLE FILE CREATED BY 'RTD' OPTION:

1. 1 BYTE = C0H. BACKWARD EOF. IGNORE IT.
2. 1 OR MORE ENTRIES OF THE FORM:
    1. 'LOCATION' LOW VALUE BYTE
    2. 'LOCATION' HIGH VALUE BYTE
    3. CONTROL BYTE, BITS 7-0 (7 IS LEFTMOST)
        - 7-6 = 11: DUMMY LAST ENTRY IN TABLE. IGNORE ALL OTHER BITS AND BYTES OF ENTRY.
        - 7-6 = 01: REFERENCE ENTRY:
            1. BITS 5-0 = 0
            2. 'LOCATION' REFERENCED BY THE ONE OR MORE FOLLOWING REFERENCOR ENTRIES.
        - 7-6 = 00: REFERENCOR ENTRY:
            1. THE INSTRUCTION AT THIS 'LOCATION' REFERENCED 'LOCATION' IN THE PREVIOUS REFERENCEE ENTRY
            2. BITS 5-0 = 00-1FH CODE INDICATING TYPE OF INSTRUCTION MAKING THE REFERENCE:
                - 0 = S
                - 1 = T
                - 2 = U
                - 3 = V
                - 4 = W
                - 5 = X
                - 6-7 UNUSED
                - 8 = P
                - 9 = L
                - A = R
                - B = 1F UNUSED

            SEE A REFERENCE TABLE LISTING FOR DEFINITIONS.

PROGRAM: 'LMOFFSET' VERSION 1.1

PROGRAM RESIDES IN 5200-5FFF.

PROGRAM FUNCTIONS AS FOLLOWS:

1. READS EITHER A TAPE-TYPE ASSEMBLY LOAD MODULE FROM TAPE OR A DISK-TYPE ASSEMBLY LOAD MODULE FROM DISC.
    1. IF SOURCE FROM DISK, PROGRAM ASKS FOR SOURCE FILESPEC IN DDDDDDDD/TTT.PPPPPPPP:D FORMAT
2. DISPLAYS:
    1. AREA INTO WHICH MODULE WILL LOAD WHEN WRITTEN TO DISK.
    2. POSSIBLE CONFLICTS WITH SYSTEM STORAGE.
    3. MODULE ENTRY POINT. IF APPENDAGE HAS BEEN APPLIED, THE ENTRY POINT WILL BE INTO THE 15 BYTE APPENDAGE.
3. ASKS FOR NEW LOAD POINT. REPLY EITHER WITH NEW LOAD POINT OR SIMPLY REPLY 'ENTER' IF SATISFIED WITH CURRENT LOAD POINT.
4. IF NEW LOAD POINT SPEC'ED, DIFFERENT FROM THE ORIGINAL LOAD POINT THE PROGRAM ASKS IF THE APPENDAGE IS TO BE SUPPRESSED. IF SO, THE RESULTING MODULE CAN ONLY BE USED VIA DOS 'LOAD' COMMAND AS THERE IS NO APPENDAGE TO MOVE THE PROGRAM TO ITS EXECUTION AREA. RESULTING OUTPUT LOAD MODULE CAN BE USED VIA 'LOAD' WHERE TWO OR MORE LOAD MODULES ARE LOADED INTO MAIN MEMORY AND THEN STORED AS ONE LOAD MODULE VIA 'DUMP'.
5. IF NEW LOAD POINT SPEC'ED, GO BACK TO #2 ABOVE.
6. IF AT LEAST ONE NEW LOAD POINT SPEC'ED AND APPENDAGE NOT SUPPRESSED, PROGRAM ASKS IF INTERRUPTS ARE TO BE DISABLED. IF REPLY = 'Y' PROGRAM INSERTS A 'DI' (DISABLE INTERRUPTS INSTRUCTION) INTO THE APPENDAGE AS ITS ONLY ATTEMPT TO PREVENT INTERRUPTS FROM OCCURRING.
7. ASKS FOR FILESPEC OF DISK-TYPE LOAD MODULE TO BE CREATED. REPLY WITH FILESPEC IN NNNNNNNN/TTT.PPPPPPPP:D FORMAT.
8. WRITES THE NEW DISK-TYPE LOAD MODULE TO DISK. IF A NEW LOAD POINT WAS SPEC'ED:
    1. THE LOAD ADDRESS FOR EACH OBJECT CODE RECORD IS ALTERED.
    2. IF APPENDAGE NOT SUPPRESSER AN EXTRA OBJECT CODE RECORD (CALLED THE APPENDAGE) IS INSERTED BEFORE THE ENTRY POINT RECORD.
    3. THE ENTRY POINT FOR THE MODULE BECOMES:
        1. IF APPENDAGE SUPPRESSED, ENTRY POINT = 0000H.
        2. OTHERWISE ENTRY POINT IS TO FIRST BYTE OF APPENDAGE.

IF THE MODULE LOAD POINT HAS BEEN CHANGED AND APPENDAGE NOT SUPPRESSED, THE MODULE IS EXTENDED BY A 15 BYTE APPENDAGE, WHICH IS EXECUTED WHEN CONTROL IS PASSED TO THE MODULE ENTRY POINT. THE APPENDAGE EXECUTION IS AS FOLLOWS:

1. EXECUTES A DISABLE INTERRUPTS INSTRUCTION, IF SPEC'ED WHEN APPENDAGE CREATED.
2. MOVES THE OBJECT CODE (LESS THE APPENDAGE) FROM THE 'NEW' LOAD AREA TO THE 'OLD' LOAD AREA.
3. PASSES EXECUTION CONTROL TO THE 'OLD' ENTRY POINT

THIS PROGRAM DOES NOT PERFORM ANY OBJECT CODE RELOCATION!!!!
IT ONLY MOVES CODE TO NEW LOAD LOCATIONS (APPLYING THE APPENDAGE) SO THAT DOS CAN LOAD THE MODULE FROM DISK AND PASS CONTROL TO IT. IF THE PROGRAM EXECUTES FROM DOS CONFLICTING AREA, IT MUST BE CAREFUL NOT TO RE-ENABLE THE INTERRUPTS.

IF THE SOURCE PROGRAM LOADS INTO THE DISPLAY AREA (3C00-3FFF) WITHOUT OVERFLOWING IT, THOSE OBJECT CODE RECORDS WILL NOT HAVE THEIR LOAD ADDRESSES MODIFIED.

4000-51FF MEMORY IS DOS. ANY PART OF THE TARGET PROGRAM EXECUTING FROM THERE WILL CAUSE DOS AND/OR TARGET PROGRAM MALFUNCTION.

5200-6FFF MEMORY IS WHERE DOS COMMANDS EXECUTE FROM. YOUR DOS FUNCTIONS-USING PROGRAM MAY HAVE ITS LOAD POINT IN THIS REGION AND DOS WILL LOAD AND EXECUTE THE PROGRAM AS A COMMAND, BUT THE DOS COMMAND 'LOAD' WILL REJECT THE MODULE.

7000-FFFF, NO PROBLEMS, OTHER THAN LACK OF RAM, FOR PROGRAM LOADING IN THIS AREA.

WARNING!!!!!!! FOR EACH TIME A GIVEN LOAD MODULE IS USED AS THE INPUT SOURCE AND A NEW LOAD POINT IS SPECIFIED, AN ADDITIONAL APPENDAGE IS APPENDED. THIS PROGRAM KNOWS AND CARES NOTHING ABOUT PREVIOUSLY EXISTING APPENDAGES.

DURING SOURCE READ, THIS PROGRAM LOADS THE OBJECT MODULE INTO A CONTROL TABLE STARTING AT LOCATION 6000H, (VIEWABLE VIA, 'DEBUG'). IF THE OBJECT MODULE IS FROM DISK, THE DATA STORED IN THE TABLE IS EXACTLY THAT FOUND ON DISK, IF THE OBJECT MODULE IS FROM TAPE, IT IS CONVERTED TO DISK OBJECT MODULE FORMAT WHEN STORED IN THE TABLE. THE TABLE HAS THE FOLLOWING LOGICAL RECORD FORMATS:

1. OBJECT CODE RECORD.
    1. ID = 01
    2. 1 BYTE COUNT OF OBJECT BYTES +2, (0 - 2 = 256 - 258)
    3. 2 BYTE LOAD ADDRESS
    4. OBJECT BYTES
2. ENTRY POINT RECORD.
    1. ID = 02
    2. 02H BYTE COUNT
    3. 2 BYTE ENTRY ADDRESS
3. COMMENT RECORD.
    1. 1 BYTE ID.
        1. ID = 0 OR 3 - FF IF MODULE FROM DISK
        2. ID = 0 OR 3 - FB IF MODULE FROM TAPE
    2. 1 BYTE COMMENT BYTE COUNT. 0 = 256.
    3. COMMENT BYTES
4. (TAPE INPUT ONLY) EXTRANEOUS PREFIX BYTES REC. THIS REC CONTAINS ALL DATA BYTES ENCOUNTERED ON THE TAPE BEFORE THE 'U' (55H) ID BYTE ENCOUNTERED.
    1. ID = FF.
    2. 2 BYTE LENGTH COUNT
    3. EXTRANEOUS BYTES
5. (TAPE INPUT ONLY) NAME RECORD. CONTAINS ALL BYTES FROM THE 'U' SYNCH BYTE TO EITHER A 3CH BYTE (ID FOR TAPE OBJECT CODE RECS) OR A 78H BYTE (ID FOR TAPE ENTRY POINT REC) ENCOUNTERED.
    1. ID = FE.
    2. 2 BYTE LENGTH COUNT
    3. NAME BYTES
6. (TAPE INPUT ONLY) IMBEDDED EXTRANEOUS DATA EEC. CONTAINS EXTRANEOUS DATE APPEARING BETWEEN END OF ONE OBJECT CODE REC AND EITHER THE NEXT OBJECT CODE RECORD OR THE ENTRY POINT RECORD.
    1. ID = FD.
    2. 2 BYTE LENGTH COUNT
    3. EXTRANEOUS DATA BYTES
7. (TAPE INPUT ONLY) BAD CHECKSUM RECORD. FOLLOWS AN OBJECT CODE RECORD WHOSE CHECKSUM COMPUTED BAD.
    1. ID = FC.
    2. 1 BYTE CHECKSUM FROM TAPE.
    3. 1 BYTE COMPUTED CHECKSUM DURING OBJECT RECORD READ.

DURING READ FROM TAPE:

1. SINGLE '*' DISPLAYED WHEN READY FOR TAPE. DO REWIND (IF NECESSARY), FAST FORWARD POSITIONING (IF NECESSARY) AND 'PLAY'
2. '***' APPEARS WHEN TAPE READ SYNCHRONIZATION HAS COMPLETED.

3. ALTERNATES '*' AND BLANK IN RIGHT-MOST DISPLAY POSITION AS EACH OBJECT RECORD READ COMPLETES WITH GOOD CHECKSUM.
4. DISPLAYS 'C' WHEN A BAD CHECKSUM ENCOUNTERED.
5. DISPLAYS 'P' IF LEADING EXTRANEOUS DATA BYTES ENCOUNTERED.
6. DISPLAYS 'I' IF IMBEDDED EXTRANEOUS DATA BYTES ENCOUNTERED.

DOCUMENTATION FOR APPARAT 'SUPERZAP' 2.0

***** WARNING - PROGRAM MALFUNCTIONS IF ANY WRITE OPERATIONS ATTEMPTED TO A WRITE-PROTECTED DISKETTE. **********

FUNCTION 'DD' DISPLAYS A DISK SECTOR. FUNCTION 'DM' DISPLAYS A MODULO 256 BYTE BLOCK OF MAIN MEMORY RESPECTIVELY. FOR 'DD', THE ERROR CODE RETURNED BY THE SYSTEM IS DISPLAYED IN COLUMN 7 OF THE LAST LINE. IF = 6, THE READ WAS SUCCESSFUL BUT THE SECTOR IS READ-PROTECTED (IE, NOT READABLE BY NORMAL USER DISK OPERATIONS, BUT WAS READ BY SUPERZAP).

FOR 'DD' AND 'PD' FUNCTIONS, COLUMNS 1-6 ARE DEFINED:

1	-	RELATIVE DISK NUMBER (0 - 3).
2-3	-	RELATIVE TRACK NUMBER (HEX 0 - 22).
4	-	RELATIVE SECTOR NUMBER ON TRACK (0 - 9.).
5-6	-	RELATIVE POSITION (HEX) WITHIN THE SECTOR.

WHEN A 'DD' SECTOR OR A 'DM' BLOCK IS DISPLAYED ON THE SCREEN, THE PROGRAM IS CONSTANTLY MONITORING THE INPUT KEYS LOOKING FOR ONE OF THE FOLLOWING COMMANDS:

'X'	-	TERMINATE THE PRIMARY FUNCTION.
'R'	-	DISPLAY AGAIN THE SAME SECTOR/BLOCK.
'+' OR ';'	-	SCROLL FORWARD ONE SECTOR/BLOCK.
'-' OR '='	-	SCROLL BACKWARD ONE SECTOR/BLOCK.
'J'	-	RESTART THE SAME PRIMARY FUNCTION.
'K'	-	'DD' ONLY. SAME AS 'J' EXCEPT NEXT SECTOR FROM SAME DRIVE AS PREVIOUS DISPLAYED SECTOR.
'MOD&&'	-	MODIFY THE CURRENT SECTOR/BLOCK. && IS THE RELATIVE POSITION WITHIN THE SECTOR/BLOCK OF BYTE TO BE MODIFIED (HEX 00 - FF).

THE PROGRAM RESPONDS BY PLACING AN 'M' IN COLUMN 7 OF THE MODIFICATION LINE. FURTHER, IT PRECEDES THE HEX DIGIT QUAD CONTAINING THE NEXT HEX DIGIT TO BE CHANGED WITH AN INDICATOR OF WHICH DIGIT IT IS, USING '+', '-', '*' AND '/' FOR THE 1ST, 2ND, 3RD AND 4TH DIGITS OF THE QUAD RESPECTIVELY.

RESPONSES WHILE IN 'MODIFICATION MODE' ARE:

HEX DIGIT 0-F	-	REPLACES CURRENT DIGIT AND ADVANCES TO NEXT DIGIT. 'DM' REPLACEMENTS ARE FOR REAL. 'DD' REPLACEMENTS ARE BUFFERED UNTIL EITHER AN 'ENTER' OR A 'Q' COMMAND. WHEN THE LAST DIGIT DISPLAYED IS REPLACED, NEXT POSITION INCREMENT WRAPS AROUND TO THE FIRST DIGITS BUT NO FURTHER DIGIT REPLACEMENTS ARE ACCEPTED UNTIL AFTER A NON-HEX-DIGIT KEY IS PRESSED.
SPACE	-	CURRENT DIGIT IS NOT CHANGED AND POSITION IS ADVANCED 1 DIGIT.
RIGHT-ARROW	-	SAME AS 'SPACE'.
SHIFT-LEFT-ARROW	-	CURRENT DIGIT NOT CHANGED AND POSITION IS RETARDED 4 DIGITS.

LEFT-ARROW - CURRENT DIGIT NOT CHANGED AND POSITION IS RETARDED 1 DIGIT.

SHIFT-RIGHT-ARROW - CURRENT DIGIT NOT CHANGED AND POSITION IS ADVANCED 4 DIGITS.

UP-ARROW - CURRENT DIGIT NOT CHANGED AND POSITION IS RETARDED 1 LINE. FIRST LINE WRAPS TO LAST.

DOWN-ARROW - CURRENT DIGIT NOT CHANGED AND POSITION ADVANCED 1 LINE. LAST LINE WRAPS TO FIRST.

'Q' - ('DD' ONLY). CANCELS WRITE OF UPDATED SECTOR. TERMINATES 'MODIFICATION MODE'.

ENTER - TERMINATES 'MODIFICATION MODE'. FOR 'DD', THE UPDATED SECTOR IS WRITTEN BACK TO DISK.

'SCOPY' - ('DD' ONLY) MOVE THE DISPLAYED SECTOR TO A DISK LOCATION TO BE SPECIFIED.

THE USER WILL BE ASKED TO RESPOND 'Y' OR 'N' AS TO THE TO-BE READ PROTECT STATE OF THE DESTINATION SECTOR.

FUNCTIONS 'PD' AND 'PM' SEND THE DISPLAYED BLOCKS TO THE PRINTER.

TO HALT PRINTER ACTIONS HOLD DOWN THE 'H' KEY UNTIL THE PRINTER STOPS. THIS ACTION TERMINATES THE FUNCTION.

FOR PRINTER OPERATIONS YOU MUST SPECIFY A BYTE COUNT (FOR 'PM') OR A SECTOR COUNT (FOR 'PD') SO PROGRAM WILL KNOW WHEN TO STOP PRINTING.

THE 'ZERO DISK SECTORS' FUNCTION DOES JUST THAT!!!!!!

IF A SECTOR BEING ZEROED IS READ-PROTECTED THE USER IS ASKED (REPLY 'Y' OR 'N') IF HE WANTS THAT SECTOR TO REMAIN READ-PROTECTED.

'COPY DISK SECTORS' FUNCTION COPIES THE SPECIFIED SECTORS TO NEW DISK LOCATIONS, DUPLICATING INTO THE NEW LOCATIONS THE READ-PROTECT STATUS OF THE OLD.

NORMALLY, THE COPY PROCEEDS IN ASCENDING TRACK/SECTOR # ORDER. HOWEVER, IF THE LOWEST DESTINATION SECTOR LIES WITHIN THE SOURCE RANGE OF SECTORS THEN THE COPY PROCEEDS IN DESCENDING TRACK/SECTOR # ORDER, COMPUTING INTERNALLY THE HIGHEST TRACK/SECTOR # OF EACH RANGE BEFORE STARTING THE COPY.

'COPY DISK DATA' ALLOWS A STRING OF DATA LESS THAN 65536 BYTES LONG TO BE COPIED FROM ANY POSITION ON THE DISKS TO ANY OTHER POSITION ON THE DISKS.

NORMALLY, THE COPY PROCEEDS IN ASCENDING TRACK/SECTOR/BYTE ORDER. HOWEVER, IF THE LOWEST DESTINATION BYTE ADDRESS LIES WITHIN THE RANGE OF THE SOURCE, THE 'COPY' PROCEEDS IN DESCENDING TRACE/SECTOR/BYTE ORDERS COMPUTING INTERNALLY THE HIGHEST TRACK/SECTOR/BYTE # OF EACH RANGE BEFORE STARTING THE COPY.

THE READ PROTECT STATUS OF DESTINATION SECTORS UPDATED BY THE COPY ARE NOT CHANGED.

'VERIFY DISK SECTORS' FUNCTION READS THE SPECIFIED SECTORS. A PAUSE OCCURS AT EACH NON-READABLE SECTOR. READ-PROTECTED SECTORS ARE IDENTIFIED AND, OPTIONALLY, PAUSES MADE.

'DISK BACKUP" COPIES THE CONTENTS OF THE SOURCE DISKETTE TO THE DESTINATION DISKETTE. ALTHOUGH VERY SLOW, IT OFTEN RETRIEVES SECTORS NOT READABLE BY REGULAR COPY OR BACKUP ROUTINES.

THE DESTINATION DISKETTE:

1. CANNOT BE ON SAME DRIVE AS SOURCE DISKETTE.
2. CANNOT BE NON-FORMATTED.
3. IS NOT TESTED FOR NAME/CONTENTS BEFORE BEING WRITTEN.
4. HAS READ-AFTER-WRITE VERIFY DONE.

IN GENERAL, WHEN INPUTTING DRIVE #, TRACK #, SECTOR #, COUNTS, ETC., KEYING AN 'X' AS THE ONLY INPUT CHAR WILL CAUSE THE FUNCTION TO BE CANCELLED.

END OF EXPLANATIONS ************